



## RAPPORT DE STAGE

---

# Assistant Ingénieur

---



Gaël SAGON  
ING3 info  
Année 2017-2018

*Tuteur : M. Alexandre Dupeyras  
Marraine : Mme. Emilie Caillault*

6 juin 2017 — 4 août 2017

RAPPORT DE STAGE  
ASSISTANT INGÉNIEUR

**SUJET :** Supervision de voies ferrées

**ÉLÈVE-INGÉNIEUR EIL Côte d'Opale :** SAGON Gaël

**ENTREPRISE :** Clemessy  
3 route de Bergues, 59210 Coudekerque-Branche  
03 28 59 57 28

## RÉSUMÉ DU STAGE ASSISTANT INGÉNIEUR

**Résumé :** J'ai effectué mon stage de deuxième année du cycle ingénieur en tant qu'assistant ingénieur à Clemessy, une société de prestation de services dans l'industrie. Pendant mon stage, j'ai pu participer au projet ULI, un projet d'automatisation des voies ferrées d'Arcelor-Mittal.

Mon stage consistait à travailler sur la supervision du projet, la partie permettant à un utilisateur de gérer les déplacements des trains. Ce projet m'a permis d'approcher deux domaines que je connaissais peu, l'automatisme et la supervision ainsi qu'un langage de programmation, le Visual Basic for Applications (VBA).

Ce stage m'a permis de découvrir l'esprit d'équipe en entreprise, de renforcer mes connaissances de l'entreprise acquises lors de mon précédent stage et de découvrir de la programmation orientée objet d'objets graphiques.

### PARRAIN ENTREPRISE :

M. Alexandre DUPYRAS  
Chef de projet

### PARRAIN EIL Côte d'Opale :

Mme. Émilie CAILLAULT

## Remerciements

Je tiens à remercier en premier lieu M. Pascal GESQUIÈRE, directeur de l'agence Clemessy de Dunkerque qui m'a accepté en stage au sein de l'entreprise.

Je remercie ensuite M. Alexandre DUPEYRAS, chef de projet, pour m'avoir encadré lors de mon stage et pour m'avoir accepté sur un de ses projets.

Ensuite, je remercie Mme. Émilie CAILLAULT, ma marraine de stage, pour m'avoir suivi et conseillé pour le bon déroulement de mon stage.

Je remercie tous les employés de Clemessy pour leur accueil chaleureux et leur aide durant mon stage, en particulier M. Julien NOËL et M. Maxime FICHAUX, ingénieurs informatiques.

Je remercie également l'EILCO pour avoir permis le bon déroulement de mon stage.

Enfin, je tiens à remercier toutes les personnes qui m'ont aidées aussi bien à l'obtention de mon stage, qu'à l'élaboration de ce rapport.

# Table des matières

<b>Résumé</b>	<b>3</b>
<b>Remerciements</b>	<b>4</b>
<b>Sommaire</b>	<b>5</b>
<b>Table des figures</b>	<b>6</b>
<b>Introduction</b>	<b>7</b>
<b>I Présentation de l'entreprise</b>	<b>8</b>
1 Fiche d'identité . . . . .	8
2 Historique . . . . .	8
3 Implantation géographique . . . . .	9
4 Activités . . . . .	9
5 Organisation . . . . .	10
6 Chiffres-clés . . . . .	10
7 Mon poste . . . . .	10
<b>II Activités</b>	<b>11</b>
1 Présentation générale . . . . .	11
2 Projet . . . . .	11
3 Logiciels . . . . .	11
3.1 InTouch . . . . .	11
3.2 ArchestrA . . . . .	13
3.3 Object Viewer et SMC . . . . .	15
4 Activités effectuées . . . . .	16
4.1 Prise en main de InTouch . . . . .	17
4.2 Macro Excel de création des fichiers AWL . . . . .	22
4.3 Prise en main de ArchestrA et des serveurs SQL . . . . .	27
4.4 Désactivation de clavier et de souris d'une Raspberry Pi 3 . . . . .	30
4.5 Création de vues pour le projet ULI . . . . .	31
5 Compte-rendu des activités . . . . .	38
6 Réflexions personnelles . . . . .	39
<b>III Synthèse</b>	<b>40</b>
1 Bilan du projet . . . . .	40
2 Bilan personnel . . . . .	40
<b>Conclusion</b>	<b>41</b>
<b>Annexes</b>	<b>42</b>
Annexe 1 . . . . .	42
Annexe 2 : . . . . .	43
<b>Bibliographie/Sitographie</b>	<b>49</b>

## Table des figures

1	Cartes de l'implantation géographique de Clemessy.	9
2	Organisation hiérarchique de Clemessy Dunkerque.	10
3	Application Manager.	12
4	Window Maker.	12
5	Window Viewer.	13
6	ArchestrA IDE.	13
7	Exemple de dérivation d'objets.	14
8	Object Viewer.	15
9	System Management Console.	16
10	Logiciel InTouch.	17
11	Modèle de conception de l'exercice 1.	18
12	Exemple de moteur éteint.	18
13	Exemple de moteur en défaut.	18
14	Exemple de résultat obtenu.	19
15	Choix des animations.	19
16	Exemple de résultat final de l'exercice 1.	20
17	Modèle de conception de l'exercice 2.	20
18	Exercice 2 : Première partie.	20
19	Exercice 2 : Deuxième partie.	21
20	Exercice 2 : Résultat final.	21
21	Exemple de macro sous Excel (Macro génération FC : partie aiguilles).	22
22	Exemple de lancement de macro sous Excel 2010.	22
23	FC à générer.	23
24	Script d'ajout du bouton de lancement de macro.	24
25	Macro génération FC : partie signaux.	25
26	Fichier de simulation.	25
27	Fichier de simulation : onglet Aide.	26
28	Serveurs du secteur Avalfer.	27
29	Serveurs utilisés lors de la supervision.	27
30	Code SQLServer de sélection d'itinéraires.	28
31	Fenêtre de variables sous ArchestrA.	29
32	Éditeur graphique de l'objet PN contenant l'objet graphique VOIE_ROUTIERE.	30
33	Onglet "Graphic Toolbox" d'ArchestrA.	31
34	Pop-up des horaires des PN.	32
35	Objet BAR.	33
36	Objets inutilisables par la supervision.	33
37	Schéma de conception du menu des travaux.	34
38	Panneau d'avertissement des travaux.	34
39	Menu des travaux.	34
40	Animation curseur sous ArchestrA.	35
41	Script de remise à zéro du panneau.	36
42	Script prédéfinis à la fermeture d'un panneau.	37
43	Phase de test du menu.	37

## Introduction

Notre deuxième année de cycle ingénieur se termine par un stage d'assistant ingénieur, afin de nous faire découvrir plus avant le monde de l'entreprise ainsi que le travail d'un ingénieur. J'ai effectué ce stage dans l'agence Clemessy de Dunkerque, une société de prestation de services, sous la tutelle de Alexandre Dupeyras, chef de projet. Mon stage s'est déroulé sur 9 semaines, du 6 juin au 4 août 2017.

Ce stage devait se dérouler sur un projet d'automatisation de déplacement de brames d'acier pour Arcelor-Mittal, mais le projet ayant été décalé à la fin d'année peu avant le début de mon stage, je me suis retrouvé sur le projet ULI pour cette même entreprise, projet d'automatisation des voies ferrées du client.

Pour présenter mon stage, je commencerai par une présentation de mon entreprise d'accueil, suivi du développement des logiciels utilisés et de mes activités avant de faire le bilan de mon stage.

# I Présentation de l'entreprise

## 1 Fiche d'identité

Clemessy est une entreprise caractérisée par :

- **Situation géographique :**
  - Siège social : Clemessy SA, 18 rue de Thann, 68200 Mulhouse, France
  - Entreprise d'accueil : Agence Clemessy Dunkerque, 3 route de Bergues, 59210 Coudekerque-Branche, France
- **Statut juridique :** Société Anonyme.
- **Domaines d'activités :** Electrotechnique, mécanique et automatisme.
- **Principaux clients :** Arcelor-Mittal, Syngenta, EDF, Auchan,...
- **Effectif :** Environ 130 employés sur le site de Dunkerque.
- **Résultat de l'année :** 665 millions d'euros en 2016, dont 21% à l'international.

## 2 Historique

- **1900** : Eugène Clemessy transforme, durant ses loisirs, un vieux moulin près de Brunstatt (68) en centrale électrique qui alimentera plusieurs communes. Pressentant l'avenir de cette source d'énergie, il fonde huit ans plus tard les établissements Clemessy.
- **1908** : Création des établissements Clemessy. Ouverture d'un petit magasin de vente de matériel électrique.
- **1926** : Réparation du premier moteur électrique pour l'usine chimique de Thann et création d'un atelier dédié à la réparation et au rebobinage.
- **1939** : André Clemessy succède à Eugène Clemessy.
- **1953** : Création de la première agence de proximité : Rethel.
- **1965** : L'aventure "export" débute (Cameroun, Tchad, Guinée, Gabon, Pakistan, URSS). C'est aussi le départ d'un long partenariat avec le CNES en Guyane, où Clemessy s'implante dès 1969 et participe aux installations de Kourou.
- **1969** : Implantation de Clemessy en Guyane et départ d'un long partenariat avec le CNES où l'entreprise participera aux installations de Kourou.
- **1971** : L'expansion de l'entreprise se confirme : 1 850 salariés et un chiffre d'affaires de 90 millions d'euros. C'est à cette époque que Clemessy intègre, dans ses offres, l'automatisme. En 1975, la participation au programme nucléaire français permet d'asseoir la notoriété de l'entreprise mulhousienne. Le Groupe compte 2 200 personnes pour un chiffre d'affaires de 137 millions d'euros.
- **1982** : L'entreprise entre dans le club très fermé des prestataires de hautes technologies pour le lanceur Ariane. Le CNES lui confie, en consortium international, l'informatique industrielle du contrôle-commande du remplissage de la fusée Ariane et de la séquence synchronisée du compte à rebours.
- **1999** : La famille Clemessy met en vente ses parts de la société. Le consortium EDFCOGEMA-SIEMENS devient actionnaire majoritaire.
- **2000** : Le groupe Game rejoint l'entreprise et enrichit ses expertises en ingénierie de maintenance, analyse industrielle et analyse vibratoire.

- **2001** : L'entreprise familiale d'Eugène Clemessy cède la place à un groupe à vocation internationale appelé à œuvrer dans de nombreux secteurs tels que l'énergie et l'environnement. Son nouvel actionnaire principal est Dalkia.
- **2004** : La Direction, pour appuyer la démarche de gestion de l'emploi et des compétences, décide de créer l'Institut Des Métiers (IDM). L'IDM accueille chaque année environ 1 000 stagiaires.
- **2006** : Poursuite du développement à l'international : Roumanie, Slovaquie, Maroc, Chine, Allemagne, Bénin, Kazakhstan, Sénégal, Suisse ...
- **2007** : Renforcement des expertises du Groupe par croissance externe. De nouvelles activités complètent les nombreuses offres : îlots robotisés, ensembles mécaniques, encollage, contrôle dimensionnel et contrôle non destructif.
- **2008** : En décembre 2008, Clemessy change d'actionnaire : Veolia a cédé le contrôle du capital de Clemessy SA à Eiffage, 8ème groupe européen de la construction et des concessions.
- **2015** : Au sein de la Division Systèmes (branche Energie) du Groupe EIFFAGE, Clemessy poursuit son développement. Avec 636 millions d'euros de chiffre d'affaires et 5000 collaborateurs, Clemessy compte plusieurs implantations à l'international et est reconnu comme un spécialiste de référence en génie électrique et génie mécanique.

### 3 Implantation géographique

Comme le montrent les cartes de la figure 1, extraites du site de l'entreprise, Clemessy est présent aussi bien en France qu'à l'étranger.

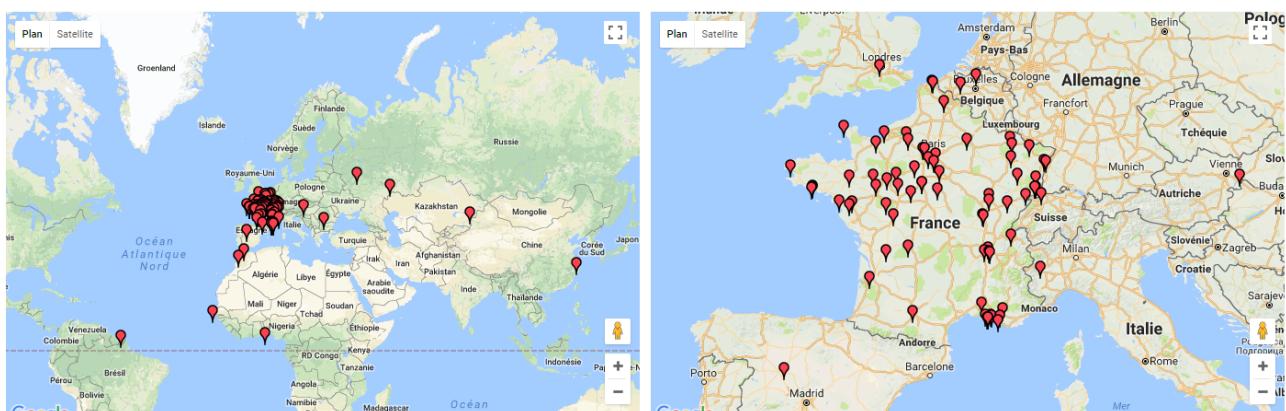


FIGURE 1 – Cartes de l'implantation géographique de Clemessy.

### 4 Activités

L'implantation géographique de l'entreprise lui permet de pouvoir effectuer aussi bien des projets de proximité que de grands projets demandant beaucoup plus de ressources, et cela dans de nombreux domaines tels que le génie thermique, le transport de fluides, le Data Center, la métallurgie et sidérurgie, la pharmacie ou la recherche nucléaire.

## 5 Organisation

À Dunkerque, Clemessy est découpée en plusieurs secteurs. De haut en bas sur la figure 2. Le secteur RH, dirigé par L.BLANCKAERT ; le secteur Achats, dirigé par P.DHEEDENE ; le secteur Qualité et Sécurité, dirigé par C.MAZUY ; les secteurs commerces Sidérurgie et Industrie ; le secteur Bureau d'Etudes dirigé par R.SINDT et le secteur Production, dirigé par Y.Clipet. Je travaillais dans le secteur Informatique du Bureau d'Etudes.

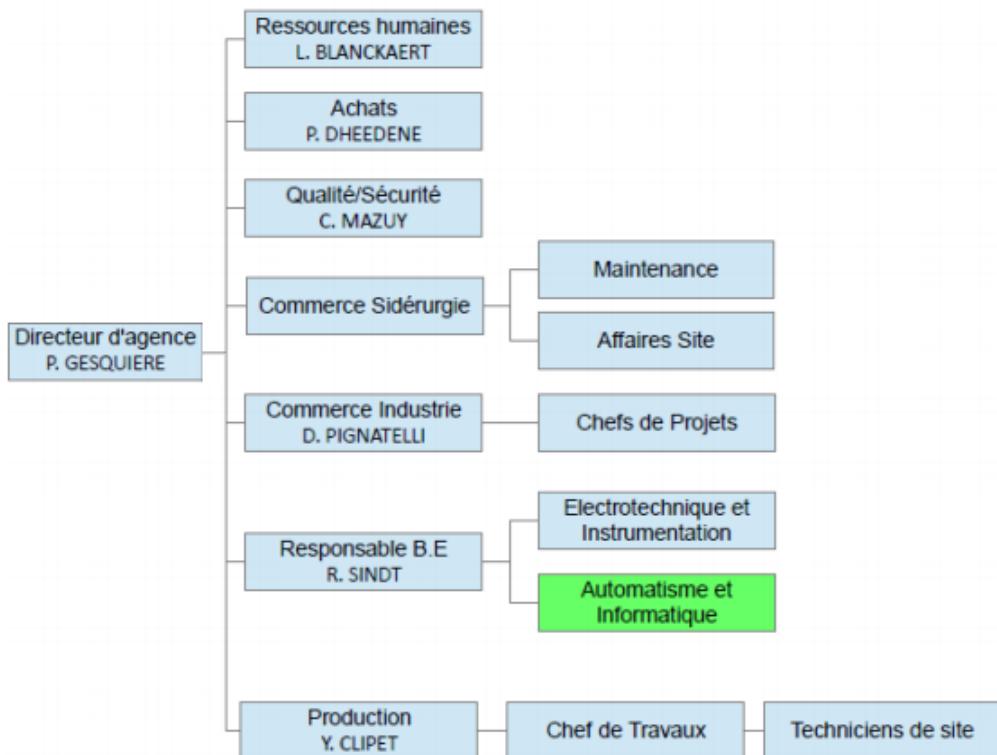


FIGURE 2 – Organisation hiérarchique de Clemessy Dunkerque.

## 6 Chiffres-clés

Comme indiqué sur le site de Clemessy, le Groupe emploie plus de 5000 collaborateurs en France et à l'international sur plus de 100 sites. C'est également 665 millions d'euros de chiffre d'affaires en 2016 dont 21% sont effectués à l'étranger.

Les commandes du Groupe sont à 58% des commandes pour l'industrie, 31% pour l'énergie et le nucléaire et 11% pour le tertiaire et les infrastructures.

## 7 Mon poste

Dans l'entreprise, je travaillais dans le secteur du Bureau d'Etudes, pour le projet ULI. J'avais pour chef de projet Alexandre DUPEYRAS, et sur le plateforme de développement, j'étais avec un automatiste et un ingénieur informatique qui travaillent en coordination sur le projet. Durant mon stage, j'ai assisté l'ingénieur informatique afin de faire une partie de son travail et de l'aider du mieux possible. Je travaillais en autonomie mais mon travail pouvait être demandé à tout moment pour le projet.

## II Activités

### 1 Présentation générale

La supervision est l'ensemble des moyens mis en œuvre pour piloter les procédés automatisés. À Clemency, la supervision est informatique et est surtout utilisée pour de la surveillance de systèmes et d'activités. Le projet ULI sur lequel j'ai travaillé consiste à l'automatisation et la supervision des voies ferrées d'Arcelor-Mittal Dunkerque. J'ai été chargé du secteur P2, "Avalfer", composé de 3 sous-secteurs : P3 (Poste 3), PN16 (Passage à niveau numéro 16), TRI (Triangle). Chaque sous-secteur dispose d'un automate qui lui est propre et qui gérera les déplacements des aiguilles et l'affichage des signaux. Le but premier de la supervision étant la gestion d'itinéraires pour les trains.

Comme je n'ai pas travaillé sur la partie automatisme car n'ayant pas les connaissances dans ce domaine, cette partie ne sera développée que brièvement.

Au cours de mon stage, mon cahier des charges variait selon mon état d'avancement. Cependant, la création de vues et d'objets graphiques animés a été ma charge principale. Afin de mieux comprendre ce que j'ai effectué pendant ces 9 semaines, je vous présenterai le projet sur lequel j'ai travaillé. Viendront ensuite les logiciels que j'ai utilisés tout au long de mon stage, puis j'expliquerai plus en détails mes activités et leurs avancements à la fin de mon stage. Je terminerai sur quelques réflexions personnelles.

### 2 Projet

Le projet ULI sur lequel j'ai travaillé donne au client la possibilité de gérer les voies, les signaux, les aiguilles des secteurs supervisés pour ainsi créer des itinéraires permettant aux trains de circuler plus rapidement et empêcher les accidents. Ces créations d'itinéraires peuvent se faire manuellement ou automatiquement. Toutes ces créations sont répertoriées dans une base de données SQL par le biais d'ArchestrA avec une commande qui permet la traduction entre le langage naturel et le langage automate. Au niveau d'ArchestrA, certaines variables servent de variables magnétoscope.

### 3 Logiciels

#### 3.1 InTouch

InTouch est un logiciel de supervision industrielle développé par Wonderware qui permet de créer des vues pour la supervision. J'ai utilisé la version 2014 R2 du logiciel, la plus récente, le plus souvent sous Windows 8. Le logiciel InTouch se compose de 3 parties :

- L'Application Manager ;
- Window Maker ;
- Window Viewer.

**Application Manager** Il s'agit de la fenêtre apparaissant à l'ouverture du logiciel. Elle permet de gérer les différentes applications InTouch installées sur le poste : création, suppression, importation ou exportation. Les deux autres parties de l'application peuvent être lancées de cette fenêtre donnée en figure 3.

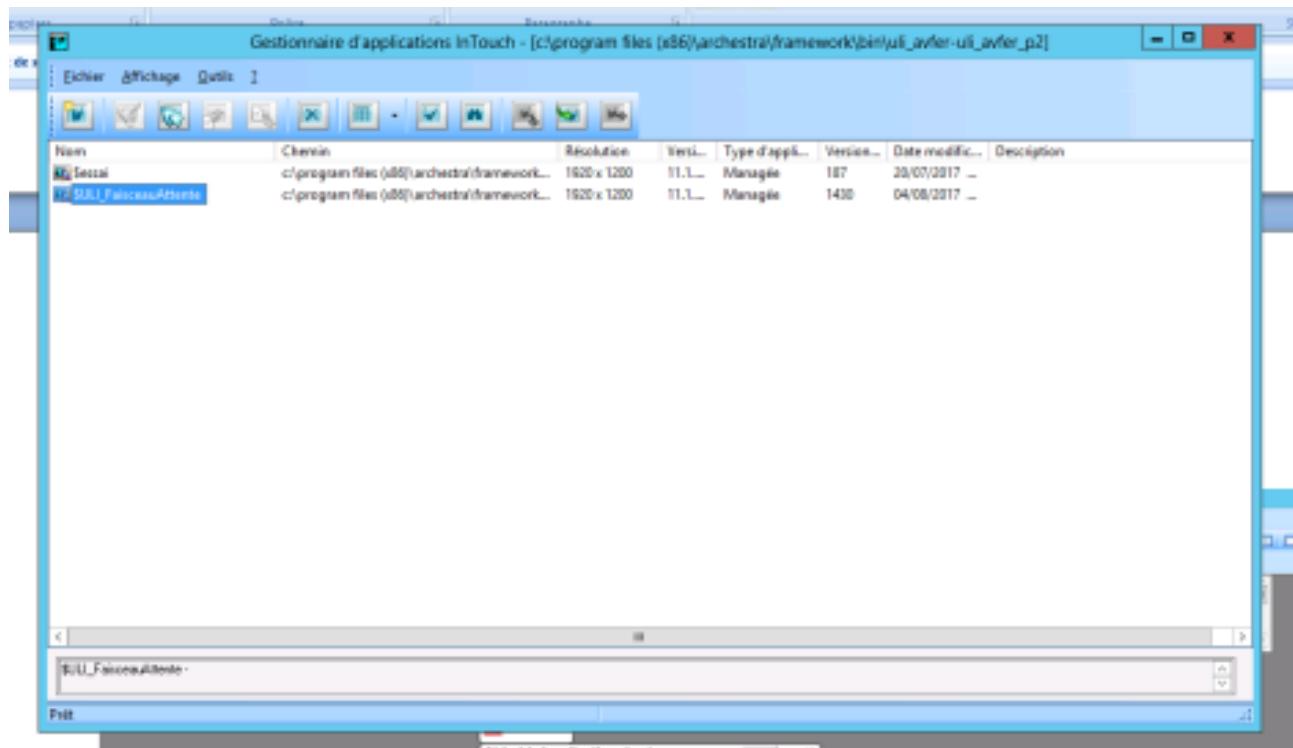


FIGURE 3 – Application Manager.

**Window Maker** Window Maker, qui est en figure 4, est la partie du logiciel permettant de développer les différentes vues de supervision. De là, il est possible d'ajouter des objets animés, des scripts, des variables et d'autres possibilités que je n'ai pas utilisées durant mon stage.

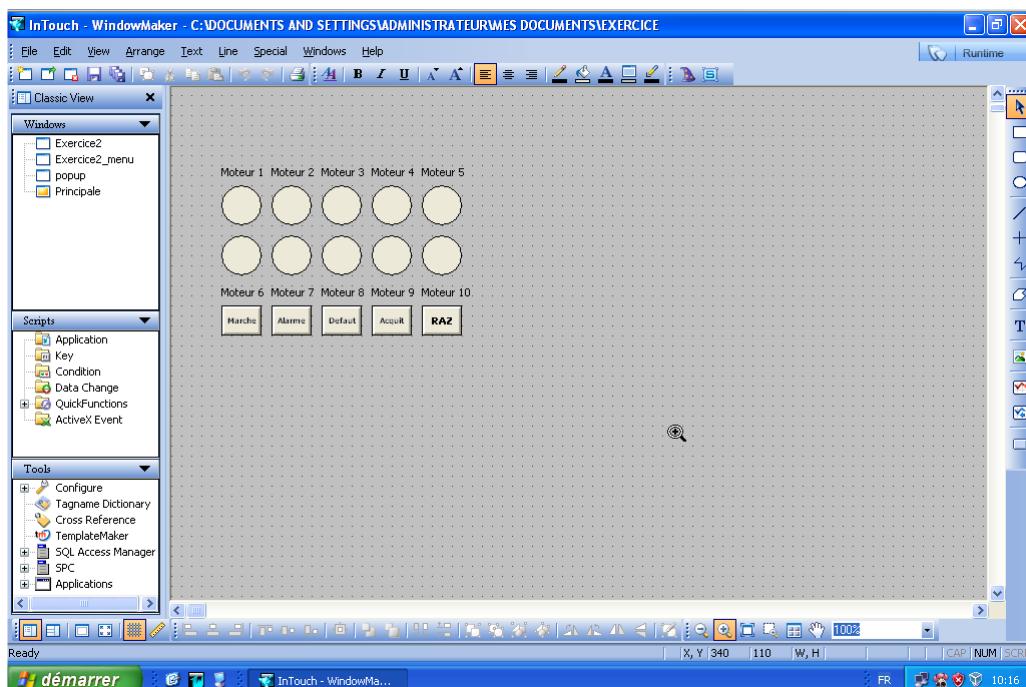


FIGURE 4 – Window Maker.

**Window Viewer** Cette dernière partie permet la visualisation et l'exploitation des vues. Durant la phase de développement, elle permet notamment de vérifier le fonctionnement des scripts et des animations. La figure 5 est un exemple de Viewer obtenu durant mon stage.

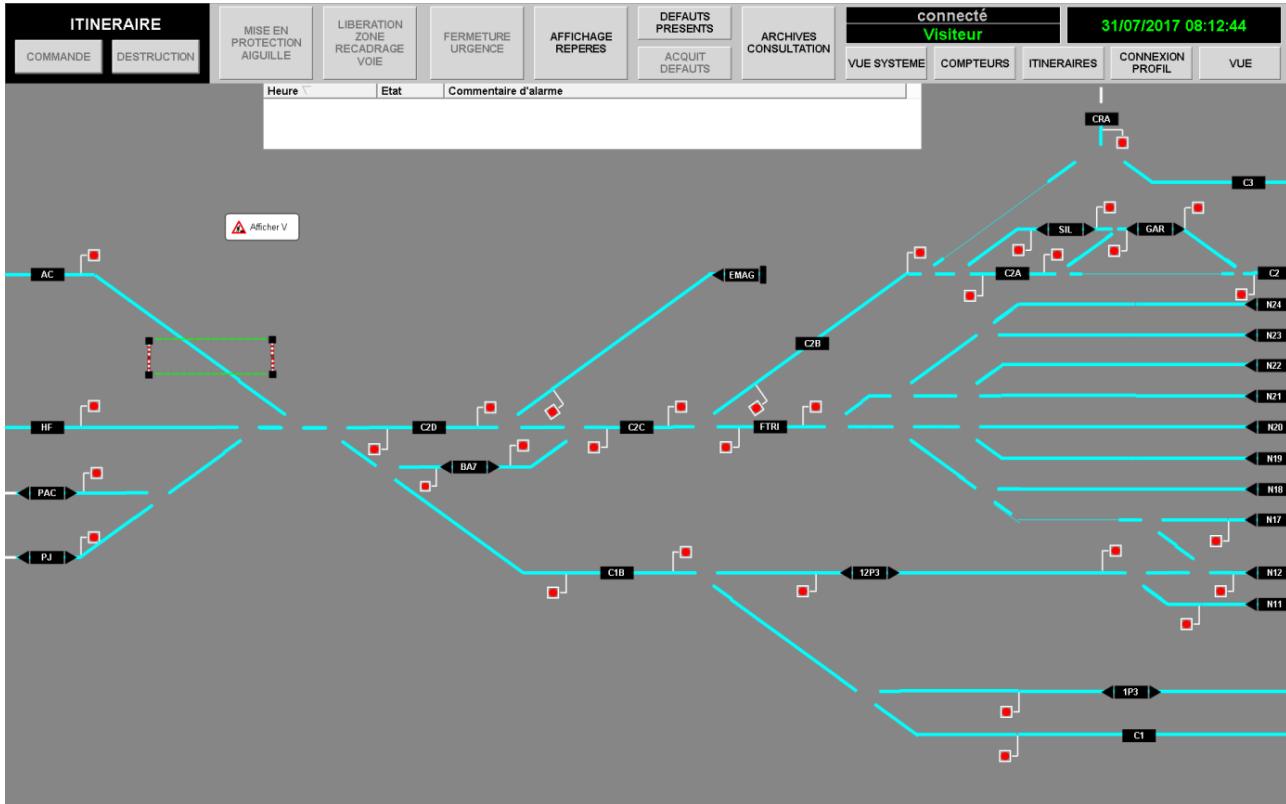


FIGURE 5 – Window Viewer.

### 3.2 ArchestrA

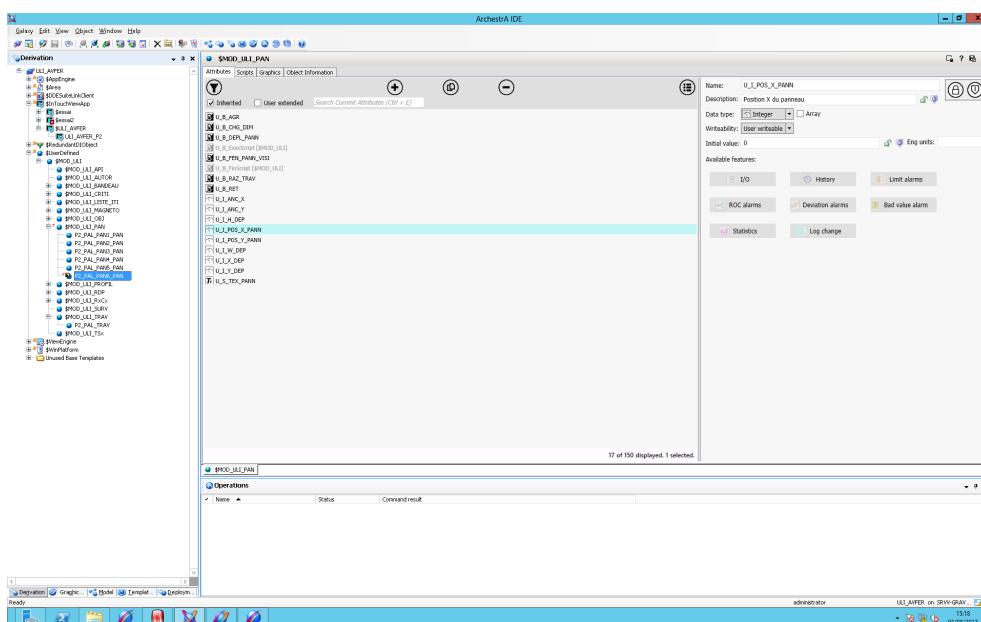


FIGURE 6 – ArchestrA IDE.

ArchestrA IDE, en figure 6, est un autre logiciel de Wonderware, plus complexe qu'InTouch et utilisant ce dernier. ArchestrA est un logiciel de programmation orientée objet. Ce logiciel permet de se connecter à une Galaxy, qui est un regroupement d'éléments tels que des aires, des objets définis par le développeur ou des applications InTouch. Le développement d'objets sous ArchestrA peut se rapprocher d'un langage de programmation orientée objet tel que JAVA. Prenons comme exemple la figure 7 ci-dessous.

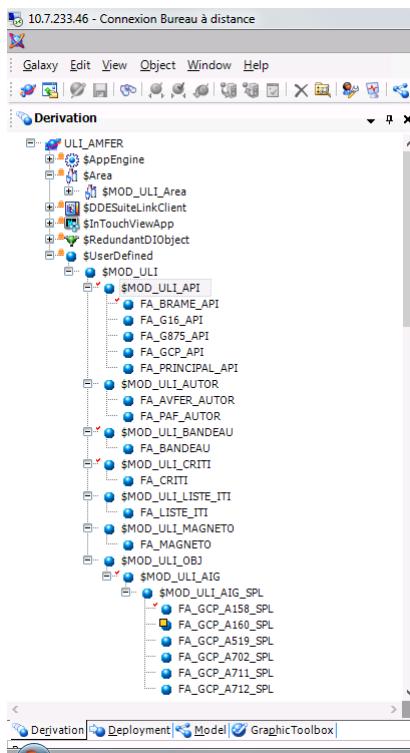


FIGURE 7 – Exemple de dérivation d'objets.

Sur cette figure 7, nous voyons le listing de dérivation de la Galaxy qui peut se traduire par un arbre d'héritage en JAVA. Intéressons-nous surtout à ce que l'on pourrait appeler en Java la classe \$UserDefined. Cette "classe", qui est un objet sous Archestra, contient les autres objets définis pour le développement du projet. Dérivée de cet objet, la "classe" \$MOD\_ULI qui contient tous les objets qui seront utilisés pour la modélisation du projet, autrement dit, l'affichage des vues. De cet objet est dérivée la "classe" \$MOD\_ULI\_OBJ qui regroupe les objets qui seront placés sur la vue principale du secteur Avalfer. Héritant de cette classe, on trouve une liste exhaustive de sous-classes telles que \$MOD\_ULI\_AIG qui représente les aiguilles, \$MOD\_ULI\_BAR qui représente les barrières, \$MOD\_ULI\_PNx qui représente les passages à niveau x, \$MOD\_ULI\_VOI qui représente les voies ferrées,... On trouve ensuite les classes \$MOD\_ULI\_AIG\_SPL, \$MOD\_ULI\_AIG\_TJS et \$MOD\_ULI\_AIG\_TJD héritant de la classe \$MOD\_ULI\_AIG. Finalement, nous trouvons des instances d'objets comme FA\_GCP\_A158\_SPL qui représente donc en JAVA un objet et qui dispose de variables, de scripts et de graphiques locaux en plus de ceux hérités. Nous verrons tout cela plus en détails peu après.

Nous avons vu qu'à la différence près de dénomination entre Classe en JAVA et Objet sous ArchestrA et entre Objet en JAVA et Instance d'Objet sous Archestra, il y a héritage ou dérivation des variables, graphiques ou scripts. Nous verrons ce qu'il en est du type des variables lors de la création du menu des travaux.

### 3.3 Object Viewer et SMC

Lors de mes tests sous ArchestrA, je me suis aidé de l’“Object Viewer” et du “SMC” (System Management Console). Le premier est une fenêtre qui permet de sélectionner les variables ArchestrA que l’on veut surveiller ou modifier facilement. C’est là où l’on peut également voir si les variables ont bien été configurées (“Quality”) et leur dernière date de mise à jour. La figure 8 montre un exemple de l’“Object Viewer”

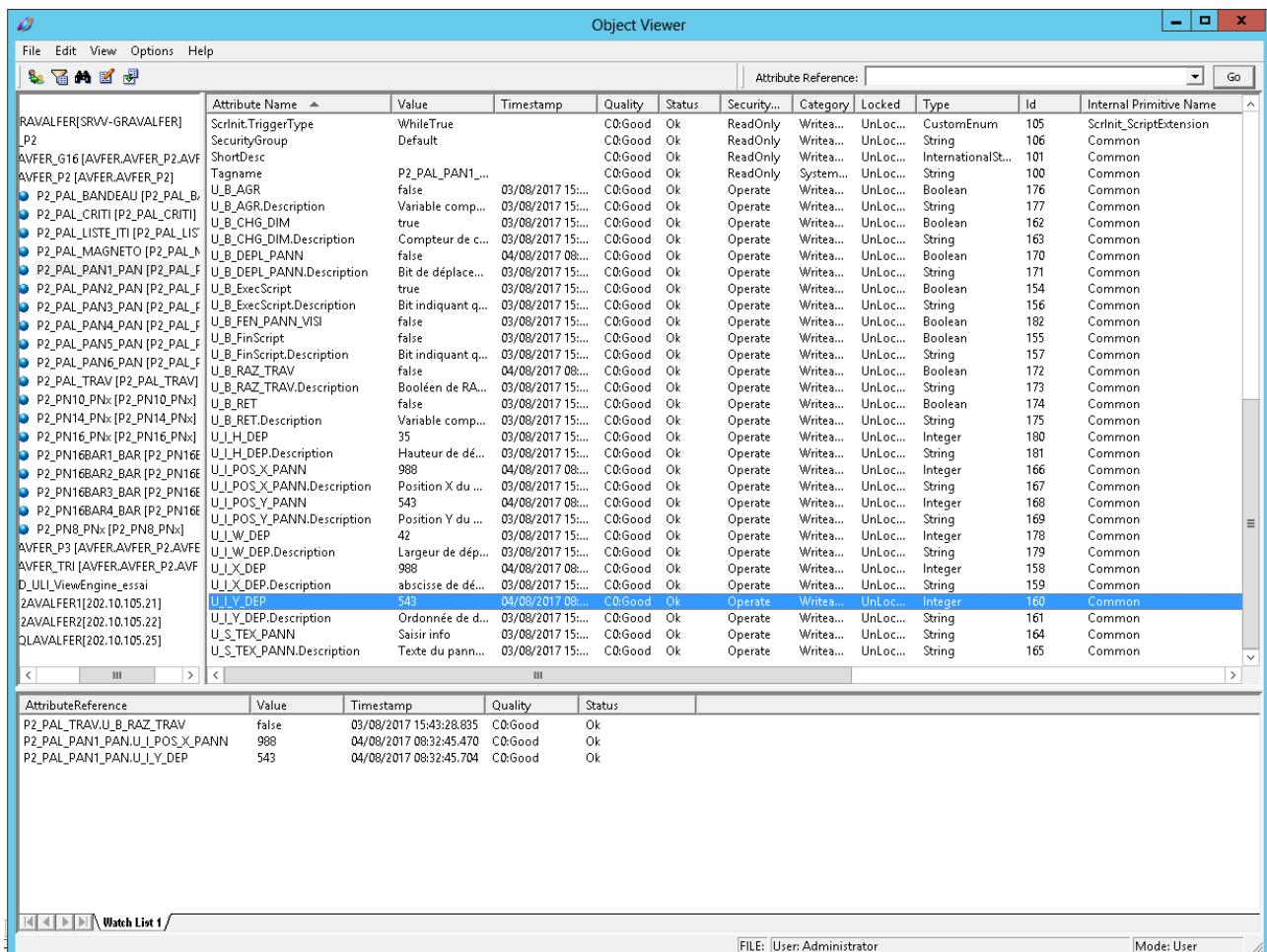


FIGURE 8 – Object Viewer.

Le second permet de voir les actions effectuées lors de la supervision. Cela peut-être des informations, en blanc, qui peuvent être demandées par le développeur avec la fonction “LogMessage()” qui va alors afficher ce qui est entré en paramètre, des avertissements, en jaune, lors d’erreurs qui n’empêchent pas la supervision de tourner ou des erreurs, en rouge, lorsque ces dernières sont critiques et empêchent le bon fonctionnement de la supervision, comme montré en figure 9.

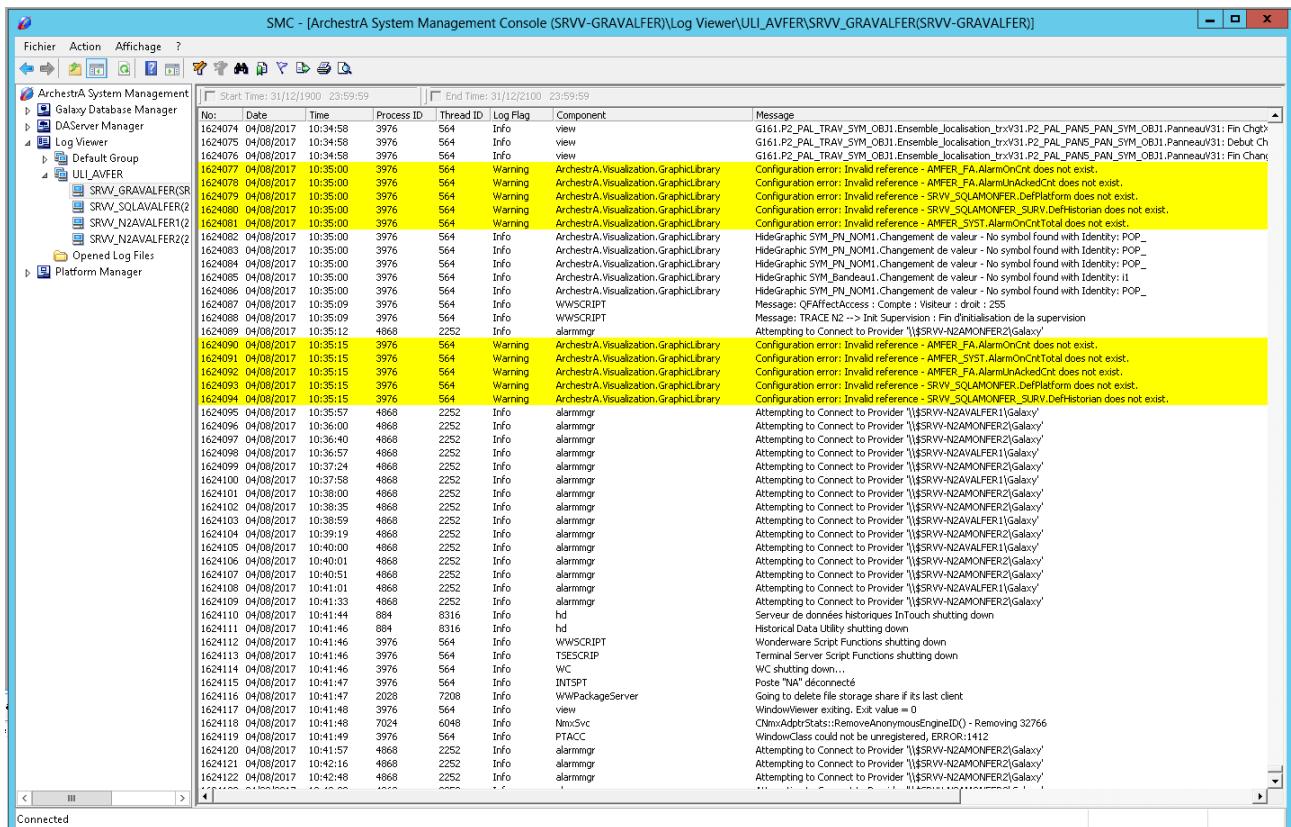


FIGURE 9 – System Management Console.

## 4 Activités effectuées

Pendant mon stage, j'ai travaillé sur la Galaxy ULI\_AVFER. Le développement de la supervision du secteur Amonfer est très proche du développement de la supervision du secteur Avalfer que ce soit par les serveurs, les objets, scripts ou procédures SQL utilisés, la Galaxy ULI\_AVFER a donc été une copie à modifier de la Galaxy ULI\_AMFER. La majorité de mon travail était donc de créer les vues, la partie visuelle de la supervision, et de modifier correctement tous les objets ArchestrA déjà créés pour l'Avalfer.

Pour cela, il fallait tout d'abord que je m'exerce sur le logiciel de supervision qu'utilise Clemessy pour la supervision : InTouch.

## 4.1 Prise en main de InTouch

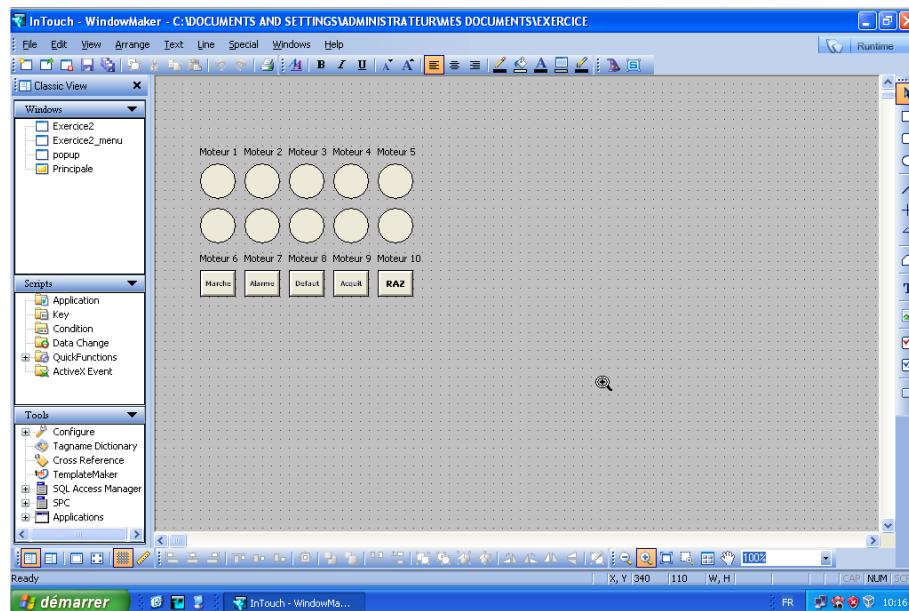


FIGURE 10 – Logiciel InTouch.

Chaque vue peut être dessinée “à la main” ou avec des objets graphiques existants. Chaque objet peut être animé de plusieurs façons, ce qui permet d’avoir une multitude de possibilités de graphiques. Pour pouvoir mieux appréhender le projet, j’ai dû effectuer des exercices basiques de développement sur InTouch.

Le premier exercice consistait dans un premier temps à créer un moteur avec un bouton Marche (variable moteur1Marche) pour le mettre en route, un bouton Alarme (variable moteur1Al) pour mettre le moteur en attente, un bouton Défaut (variable moteur1Def) pour arrêter le moteur seulement s’il est en fonctionnement, un bouton Acquit pour mettre le moteur en attente après un défaut et un bouton RAZ pour remettre le moteur à zéro. Le moteur, représenté par un cercle, doit changer de couleur selon son état :

- vert lorsqu'il est en fonctionnement,
- jaune lorsqu'il est en attente,
- rouge lorsqu'il est arrêté,
- gris lorsqu'il est éteint.

Ce sont les phases 1 à 3 du schéma de conception de l’exercice en figure 11. Des exemples de moteur sont en figures 12 et 13 ci-dessous.

Dans un deuxième temps, phases 4 et 5, il fallait pouvoir sélectionner 1 moteur sur un certain nombre et faire apparaître une fenêtre pop-up qui permet de sélectionner l’état dans lequel mettre le moteur choisi. La fenêtre pop-up étant la même pour tous les moteurs, il fallait trouver un moyen pour que les changements de valeur sur la pop-up ne se répercutent pas sur tous les moteurs car ce sont les boutons qui ne sont pas situés sur la pop-up qui mettent tous les moteurs dans le même état. Pour cela, j’ai dû utiliser les variables indirectes qui sont des variables permettant de ne modifier que le moteur voulu. Le moteur de la pop-up est une simple copie du premier moteur avec modification du nom des variables en remplaçant “moteur1” par “popup”.

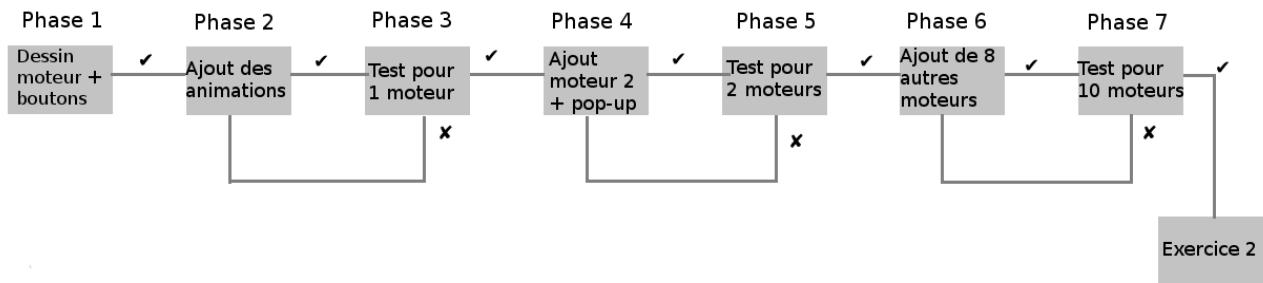


FIGURE 11 – Modèle de conception de l'exercice 1.

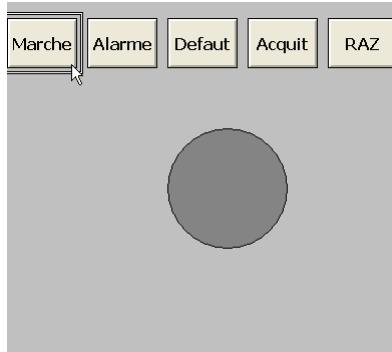


FIGURE 12 – Exemple de moteur éteint.

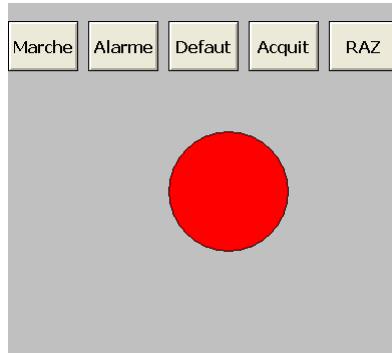


FIGURE 13 – Exemple de moteur en défaut.

Il en est de même avec les 9 autres moteurs, phases 6 et 7, en remplaçant seulement les numéros de moteur. Les variables indirectes utilisées, nommées “indMarche”, “indAl” et “indDef”, vont prendre les valeurs correspondantes du moteur choisi et à l'affichage de la pop-up, les variables de la pop-up vont être mises à jour par “popupMarche.value=indMarche.value”. Ainsi, seul le moteur choisi est modifié. La figure 14 ci-dessous donne un exemple de fonctionnement de la supervision.

Les animations des moteurs et des boutons sont choisies sur le menu en figure 15. Les animations sont de plusieurs types :

- Entrée de l'utilisateur (User Input)
- Curseur (Sliders)
- Action lors d'un clic (Touch Pushbuttons)
- Changement de couleur de lignes (Line Color)
- Changement de taille (Object Size)

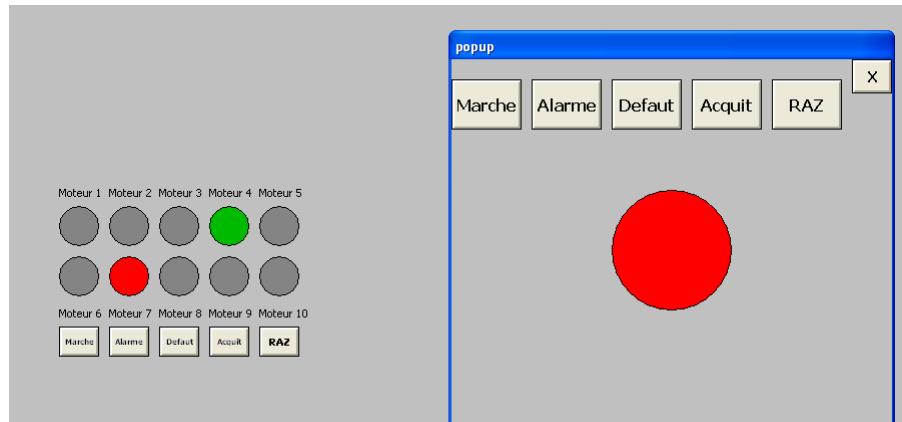


FIGURE 14 – Exemple de résultat obtenu.



FIGURE 15 – Choix des animations.

- Changement de couleur de fond (Fill Color)
- Changement de position (Location)
- Affichage de valeurs (Value Display)
- Changement de couleur de texte (Text Color)
- Remplissage selon un certain pourcentage (Percent Fill)

Dans le cas de l'exercice, j'ai surtout utilisé l'animation "Touch Pushbuttons - Action" qui consiste en l'exécution d'un script et "Fill Color - Analog" qui permet de choisir une couleur de remplissage de l'objet selon des valeurs définies. Ici, en l'occurrence, la valeur 0 donnait du gris, la valeur 1 donnait du vert, le 2 du jaune et le 3 du rouge. Le script d'action était simplement l'initialisation de la variable utilisée pour le remplissage du fond selon les conditions imposées par l'exercice. À savoir : Défaut que si le moteur est en fonctionnement, idem pour l'alarme, Acquit que si il y a un Défaut.

Le résultat final est donné par la figure 16 ci-dessous.

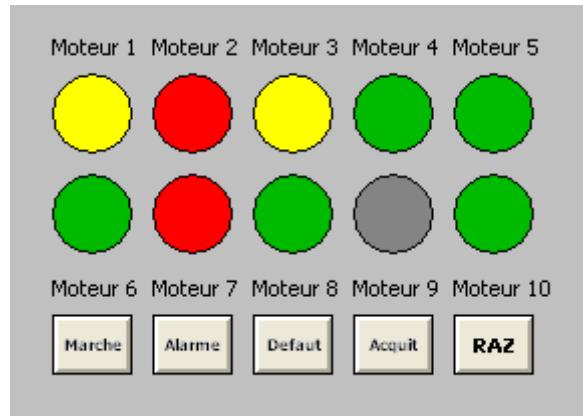


FIGURE 16 – Exemple de résultat final de l'exercice 1.

Le deuxième exercice consistait en la modélisation d'une pompe et de 2 cuves. Dans un premier temps, le remplissage des cuves se gérait grâce à un curseur qui indique le pourcentage de remplissage de chaque cuve. Le transvasement de la première cuve dans la deuxième n'étant possible que si la vanne est ouverte, la pompe allumée et la cuve 2 vide. La pompe ne pouvant pas s'allumer tant que la vanne n'est pas ouverte. Le schéma de conception est en figure 17.

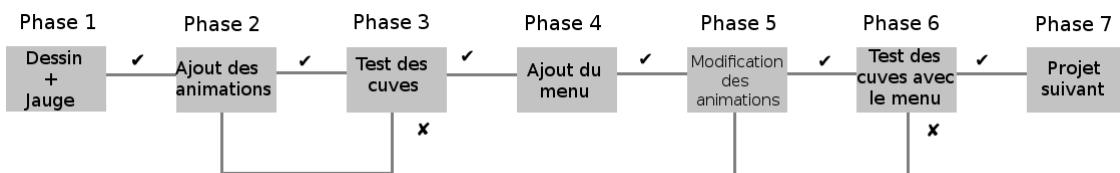


FIGURE 17 – Modèle de conception de l'exercice 2.

Les cuves sont 3 traits simples et le niveau de la cuve est un rectangle de hauteur la valeur du curseur correspondant. La vanne et la pompe ont été dessinées à la main et les curseurs et les boutons sont des objets prédéfinis sur InTouch.

Les animations utilisées sont des "Touch Pushbuttons - Discrete Value" et "Object Size - Height". La figure 18 montre son fonctionnement.

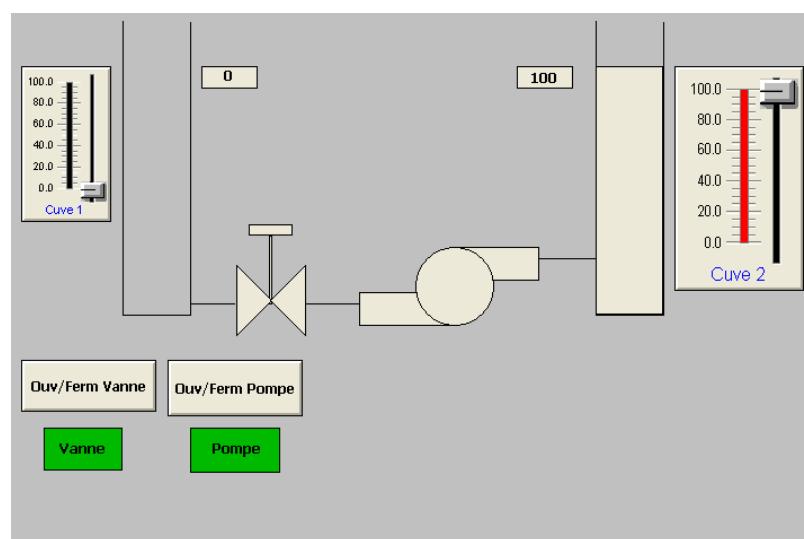


FIGURE 18 – Exercice 2 : Première partie.

Dans un deuxième temps, je devais améliorer la vue en créant un menu afin de visualiser le transvasement sur un temps de cycle total en secondes, avec démarrage de la pompe et de la vanne à des temps définis, avec les mêmes contraintes que précédemment. Le bouton Cycle lance le chronomètre, le bouton Stop Cycle met en pause le chronomètre et le bouton RAZ remet la modélisation à son état initial.

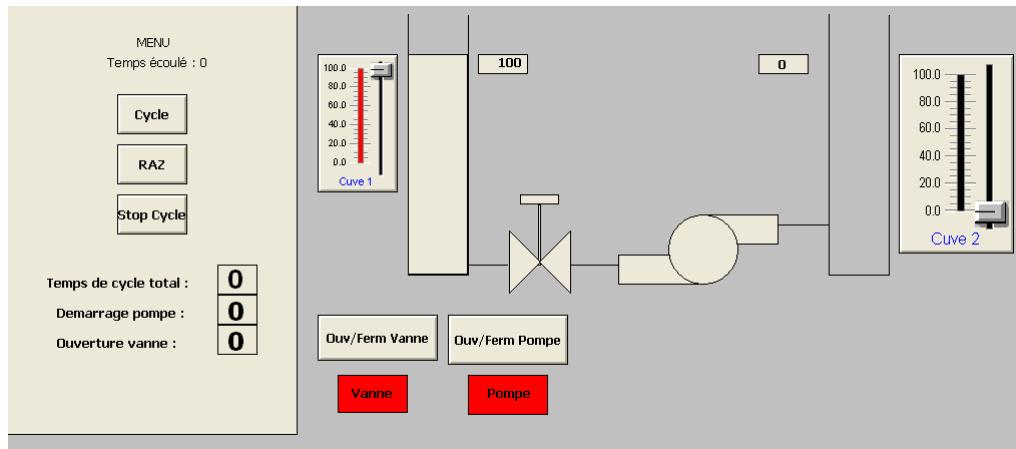


FIGURE 19 – Exercice 2 : Deuxième partie.

Pour cela, j'ai dû utiliser des "scripts de fenêtres" qui sont des scripts périodiques de période variable, ici 1000ms. Ces scripts consistent au rafraîchissement de la fenêtre après exécution du script sous la condition "While Showing" ou lors de l'ouverture ou de la fermeture de la fenêtre. J'ai appris plus tard que cette solution est plus lourde et est plus risquée que les scripts Condition ou Data Change que nous verrons après. Le langage utilisé peut être approché à du Visual Basic. Au final, la modélisation permet de voir comment évoluent les niveaux des cuves à débit constant sur une durée déterminée par l'utilisateur. La figure 20 montre le résultat obtenu.

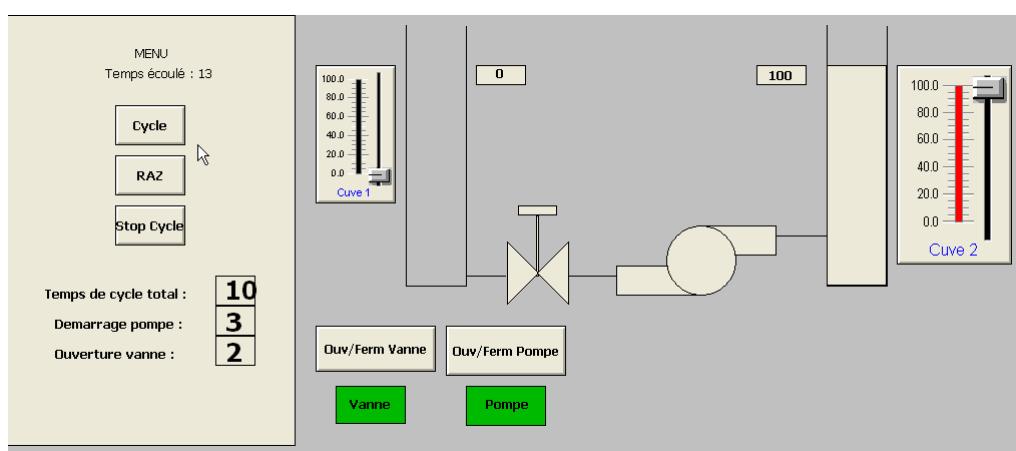


FIGURE 20 – Exercice 2 : Résultat final.

## 4.2 Macro Excel de création des fichiers AWL

Avant de me lancer concrètement sur le projet, le chef de projet m'a fourni l'AFD (Analyse Fonctionnelle Détailée) ou cahier des charges du projet sur lequel je me suis penché afin d'en prendre connaissance et de prendre rapidement en main le vocabulaire lié au projet. Afin de couper la lecture des 323 pages du cahier des charges, on m'a donné à faire des macros sous Excel.

Une macro est un script, un module du classeur courant, que l'on peut exécuter dans la barre d'outils sous Excel et qui est en VBA (Visual Basic for Applications) comme le montre les 2 figures suivantes.

```

Sub generer_FC()
    Dim ligne As String 'Définition des variables
    Dim numLigne As Integer
    Close

    ThisWorkbook.Sheets("Index").Activate ' Sélectionne la page Index

    '*****Pour aiguilles*****'

    chemin1 = ThisWorkbook.Path & "\"
    If Len(Dir(chemin1 & "FC_Aiguilles\", vbDirectory)) = 0 Then
        MkDir (chemin1 & "FC_Aiguilles\")
    End If

    chemin = chemin1 & "FC_Aiguilles\"

    nbLigne = Range("A1").End(xlDown).Row ' Donne l indice de la dernière case remplie
    For i = 2 To nbLigne

        numFC = ThisWorkbook.ActiveSheet.Range("A" & i).Value ' Donne la valeur de la case active
        numDB = ThisWorkbook.ActiveSheet.Range("B" & i).Value
        numDec = ThisWorkbook.ActiveSheet.Range("C" & i).Value
        numAig = ThisWorkbook.ActiveSheet.Range("D" & i).Value
        Open chemin1 & "FC50Test.AWL" For Input As #1 'Ouvre un fichier de lecture
        Open chemin & "FC" & numFC & ".AWL" For Output As #1 'Ouvre un fichier d'écriture (Output) ayant pour num i

        Do While Not EOF(1) ' parcours du doc de référence jusqu'à la fin
            Line Input #1, ligne
            nvLigne = Replace(ligne, ":= 50,", ":= " & numDB & ",")
            nvLigne = Replace(nvLigne, "50,", numFC & ",") 'Remplacement du numéro de FC
            nvLigne = Replace(nvLigne, "158", numAig) 'Remplacement du num de l'aiguille
            nvLigne = Replace(nvLigne, "AIG_1,", numDec & ",")
            nvLigne = Replace(nvLigne, "AIG_1", numDec & ")")
            Print #1, nvLigne
        Loop

        Close #1
        Close #i
    Next

```

FIGURE 21 – Exemple de macro sous Excel (Macro génération FC : partie aiguilles).

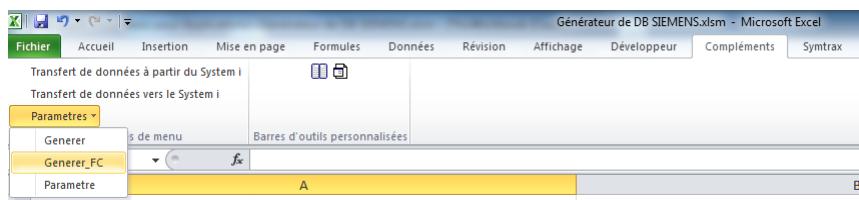


FIGURE 22 – Exemple de lancement de macro sous Excel 2010.

**Création de fichiers AWL** On m'a demandé de créer une macro qui permette la création de fichier de format AWL, lesquels seraient importés dans un logiciel au niveau automatisme. Un exemple de fichier AWL est donné en annexe.

Le but de la macro est de créer automatiquement des fichiers AWL nommés selon le format "FCX.AWL" où X est la valeur de FC de l'organe. FC signifiant Fonction. Lors de l'appel des fichiers par l'automate, chaque fichier donnera un objet différent avec des paramètres précis exploitables par le robot. Le robot est un logiciel permettant la simulation du projet.

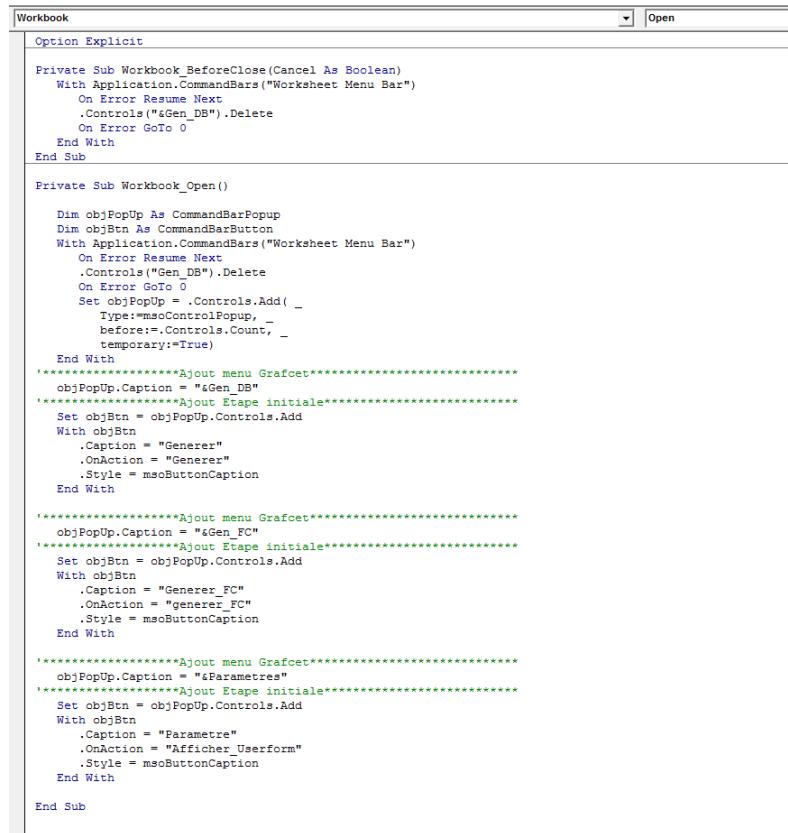
<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>E</i>	<i>F</i>	<i>G</i>	<i>H</i>	<i>I</i>	<i>J</i>	<i>K</i>	<i>L</i>	<i>M</i>
FC	NUM_DB_A	Dec	Aiguille (9)	FC	NUM_DB_A	Dec	Signaux (16)	FC	NUM_DB_A	Dec	Hex	CE (28)
1				110	110	1	S135G	10	10	1	1	CE135G
2	50	DB1	1	135				11	11	2	2	CE135D
3	51	DB2	2	142				12	12	3	3	CE142P
4	52	DB3	3	143	112	112	3	S142G	13	13	4	CE142G
5	53	DB4	4	146	113	113	4	S142P	14	14	5	CE142D
6	54	DB5	5	147	114	114	5	S143P	15	15	6	CE143G
7	55	DB6	6	148	115	115	6	S143G	16	16	7	CE143P
8	56	DB7	7	154	116	116	7	S146G	17	17	8	CE146P
9	57	DB8	8	155	117	117	8	S146D	18	18	9	CE146G
10	58	DB9	9	165	118	118	9	S147G	19	19	10	A CE146D
11					119	119	10	S147D	20	20	11	B CE147P
12					120	120	11	S148G	21	21	12	C CE147G
13					121	121	12	S148D	22	22	13	D CE147D
14					122	122	13	S154G	23	23	14	E CE148P
15					123	123	14	S155D	24	24	15	F CE148G
16					124	124	15	S165P	25	25	16	CE148D
17					125	125	16	S165P	26	26	17	11 CE154P
18									27	27	18	12 CE154G
19									28	28	19	13 CE154D
20									29	29	20	14 CE155P
21									30	30	21	15 CE155G
22									31	31	22	16 CE155D
23									32	32	23	17 CE165P
24									33	33	24	18 CE165G
25									34	34	25	19 CE165D
26									35	35	26	1A CE_PJ
27									36	36	27	1B CE_HF
28									37	37	28	1C CE_AC
29												

FIGURE 23 – FC à générer.

J'ai commencé par faire la macro pour les aiguilles (voir figure 23), donc un numéro de FC allant de 50 à 58. J'avais à disposition le fichier AW50Test.AWL qui m'a servi de modèle. La macro évite alors beaucoup de "copié-reEMPLACÉ" de ce modèle avec un risque de faute.

J'ai conçu la macro de telle manière à ce qu'elle compte le nombre d'aiguilles présentes sur la feuille Excel et que, pour chaque aiguille, elle récupère son numéro de FC, son numéro de la base de données, sa référence décimale et son numéro. Ensuite, la macro lit le fichier modèle, ouvre un fichier d'écriture nommé FCX.AWL et pour chaque ligne du fichier modèle, elle remplace les valeurs de base par les valeurs stockées peu avant, puis écrit la ligne sur le fichier d'écriture. Ensuite, elle ferme les 2 fichiers et passe à l'aiguille suivante. Le code est en figure 21.

Après avoir vérifié que tous les fichiers aient bien été créés et que les changements ont bien été faits, j'ai créé un lien dans la barre d'outils Excel, en figure 22. Ce lien est créé également en VBA, mais non pas comme module, mais directement sur le classeur, à son ouverture. Pour afficher l'option "Generer\_FC" dans la barre d'outils, il faut ajouter un bouton sur le menu qui réfère à la macro, comme montré sur la figure 24.



```

Workbook
Option Explicit

Private Sub Workbook_BeforeClose(Cancel As Boolean)
    With Application.CommandBars("Worksheet Menu Bar")
        On Error Resume Next
        .Controls("4Gen_DB").Delete
        On Error GoTo 0
    End With
End Sub

Private Sub Workbook_Open()
    Dim objPopUp As CommandBarPopup
    Dim objBtn As CommandBarButton
    With Application.CommandBars("Worksheet Menu Bar")
        On Error Resume Next
        .Controls("Gen_DB").Delete
        On Error GoTo 0
        Set objPopUp = .Controls.Add(
            Type:=msoControlPopup,
            before:=.Controls.Count,
            temporary:=True)
        End With
        ****Ajout menu Graftet*****
        objPopUp.Caption = "4Gen_DB"
        ****Ajout Etape initiale*****
        Set objBtn = objPopUp.Controls.Add
        With objBtn
            .Caption = "Generer"
            .OnAction = "Generer"
            .Style = msoButtonCaption
        End With
        ****Ajout menu Graftet*****
        objPopUp.Caption = "4Gen_FC"
        ****Ajout Etape initiale*****
        Set objBtn = objPopUp.Controls.Add
        With objBtn
            .Caption = "Generer_FC"
            .OnAction = "generer_FC"
            .Style = msoButtonCaption
        End With
        ****Ajout menu Graftet*****
        objPopUp.Caption = "&Parametres"
        ****Ajout Etape initiale*****
        Set objBtn = objPopUp.Controls.Add
        With objBtn
            .Caption = "Parametre"
            .OnAction = "Afficher_Userform"
            .Style = msoButtonCaption
        End With
    End Sub

```

FIGURE 24 – Script d'ajout du bouton de lancement de macro.

Lorsque tout cela fonctionna, j'ai dû en faire de même pour les signaux. Pour séparer les aiguilles des signaux, on m'a demandé de créer deux dossiers FC\_AIG et FC\_SIG et de générer tous les fichiers en même temps. Donc, dans la même fonction de la macro, je devais rajouter la partie pour les signaux pour lesquels j'avais également un modèle, le fichier FC110Test.AWL. Le seul changement était que les signaux nécessitaient 2 variables "DB\_MEM\_FRONT" devant valoir 0 et 1 pour le premier signal, 2 et 3 pour le deuxième signal, et ainsi de suite. Le fonctionnement de cette deuxième partie de macro est le même que pour les aiguilles (aux changements de colonnes et de valeurs à remplacer près) sauf pour une partie du modèle, de la ligne 78 à 118, qui n'était pas modifiable autrement qu'à la main donc laissée intacte. Le code final de la partie des signaux est donné en figure 25.

```

*****Pour signaux*****
If Len(Dir(chemini & "FC_Signaux\", vbDirectory)) = 0 Then
    MkDir (chemini & "FC_Signaux\")
End If
chemin = chemin & "FC_Signaux"
nbLigne = Range("E1").End(xlDown).Row

For i = 2 To nbLigne
    numLigne = 0
    numFC = ThisWorkbook.ActiveSheet.Range("E" & i).Value
    numDB = ThisWorkbook.ActiveSheet.Range("F" & i).Value
    numDec = ThisWorkbook.ActiveSheet.Range("G" & i).Value
    sig = ThisWorkbook.ActiveSheet.Range("H" & i).Value
    memFront1 = 2 * (numDec - 1)
    memFront2 = memFront1 + 1
    numSig = Mid(sig, 2, 3)
    posSig = Right(sig, 1)
    Open chemin & "FC110Test.AWL" For Input As #1
    Open chemin & "FC" & numFC & ".AWL" For Output As i
    Do While Not EOF(1)
        numLigne = numLigne + 1
        Line Input #1, ligne
        If numLigne < 78 Or numLigne > 118 Then
            nvLigne = Replace(ligne, ":= 110,", ":= " & numDB & ",")
            nvLigne = Replace(nvLigne, "SF15BP", "SF" & numSig & posSig)
            nvLigne = Replace(nvLigne, "S15BP", sig)
            nvLigne = Replace(nvLigne, "S_15BP", "S_" & numSig & posSig)
            nvLigne = Replace(nvLigne, "S15_B_P", "S" & numSig & "_" & posSig)
            nvLigne = Replace(nvLigne, "A_15B", "A" & numSig)
            nvLigne = Replace(nvLigne, "A15_15B", "Aig" & numSig)
            nvLigne = Replace(nvLigne, "SIG_1", "SIG" & numDec & ",")
            nvLigne = Replace(nvLigne, "SIG_1;", "SIG" & numDec & ";")
            nvLigne = Replace(nvLigne, "SIG_1)", "SIG" & numDec & ")")
            nvLigne = Replace(nvLigne, "15B", " " & numSig & "_")
            If memFront1 < 10 Then
                nvLigne = Replace(nvLigne, "Mem_Front_014", "Mem_Front_00" & memFront1)
            ElseIf memFront1 < 100 Then
                nvLigne = Replace(nvLigne, "Mem_Front_014", "Mem_Front_0" & memFront1)
            Else
                nvLigne = Replace(nvLigne, "Mem_Front_014", "Mem_Front_" & memFront1)
            End If
            If memFront2 < 10 Then
                nvLigne = Replace(nvLigne, "Mem_Front_015", "Mem_Front_00" & memFront2)
            ElseIf memFront2 < 100 Then
                nvLigne = Replace(nvLigne, "Mem_Front_015", "Mem_Front_0" & memFront2)
            Else
                nvLigne = Replace(nvLigne, "Mem_Front_015", "Mem_Front_" & memFront2)
            End If
            Print #i, nvLigne
        Else
            Print #i, ligne
        End If
    Loop
    Close #1
    Close #i
Next
End Sub

```

FIGURE 25 – Macro génération FC : partie signaux.

**Création du fichier de simulation** Après que je me suis exercé en VBA et que j'ai assimilé les éléments les plus importants du cahier des charges, j'ai récupéré le travail d'un stagiaire qui était sur la création d'un fichier Excel pour la simulation du projet. Dans le fichier créé, une macro permettra d'exporter les données sous un format CSV afin de les intégrer dans le robot. Le fichier final doit comporter sur chaque onglet, le modèle de chaque même onglet du fichier modèle autant de fois qu'il y a d'organes par onglet. Un extrait du fichier final est en figure 26.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	A	PL_011_HF_VOI	AVFER_PN18		P2_PN18	AC	None															
2	X	Identifiant	Libellé		E/S	Adressé	Instruction/ Mise à l'échelle	Type	Catégorie	Prise	Prise	Prise	Prise	MIN API	MAX API	MIN physique	MAX physique	Opérations	Periode	Int (Automatique)	Libellé compact (automatique)	Taille Libellé
3	X	PL_011_HF	Mod d'état		Sort	0	I / T5 ADDRESS	MC : Mod d'état	15													
4	X	ET_BNC	Etat encodeur		Sort	0	I / B ET_BNC	INF AC : Etat encodeur	24													
5	X	ET_DCC	Etat encodeur		Sort	0	I / B ET_DCC	INF AC : Etat encodeur	21													
6	X	ET_MDC	Etat encodeur en protection		Sort	0	I / B ET_MDC	INF AC : Etat encodeur en protection	22													
7	X	ET_ADDRESS	Modèle Marconi de I / T5 ADDRESS		Modèle Marconi de I / T5 ADDRESS	0	M / I T5 ADDRESS	(MC : Valeur Marconi de I / T5 ADDRESS)	39													
8	X	ET_BNC_DR	Modèle Marconi de I / B ET_BNC		Modèle Marconi de I / B ET_BNC	0	M / B ET_BNC_DR	(MC : Valeur Marconi de I / B ET_BNC)	36													
9	X	ET_BNC_DA	Modèle Marconi de I / B ET_DCC		Modèle Marconi de I / B ET_DCC	0	M / B ET_BNC_DA	(MC : Valeur Marconi de I / B ET_DCC)	35													
10	X	ET_DCC_DR	Modèle Marconi de I / B ET_MDC		Modèle Marconi de I / B ET_MDC	0	M / B ET_DCC_DR	(MC : Valeur Marconi de I / B ET_MDC)	34													
11	X	CMD RECADRAIS VOIE	Commande de redressement		Sort	0	O / B CMD RECADRAIS VOIE	EXO AC : Commande de redressement	30													
12	X	NEW_CPT_CE	Valeur de redressement du compteur voie		Sort	0	O / I NEW_CPT_CE	MC : Valeur de redressement du compteur voie	41													
13	X	OPT_CE	Valeur du compteur voie		Sort	0	O / I OPT_CE	MC : Valeur du compteur voie	28													
14	X	OPT_DA	Valeur du compteur voie		Sort	0	M / I OPT_DA	MC : Valeur du compteur voie	24													
15	X	CMD RECADRAIS VOIE	Commande de redressement		Sort	0	O / B CMD RECADRAIS VOIE	EXO AC : Commande de redressement	30													
16	X	NEW_CPT_CE	Valeur de redressement du compteur voie		Sort	0	O / I NEW_CPT_CE	MC : Valeur de redressement du compteur voie	41													
17	X	OPT_CE	Valeur du compteur voie		Sort	0	O / I OPT_CE	MC : Valeur du compteur voie	28													
18	X	OPT_DA	Valeur du compteur voie		Sort	0	M / I OPT_DA	MC : Valeur du compteur voie	24													
19	X	VOR_INTER_API	Vvoie déclenchée sur deux automates		Sort	0	K / B VOR_INTER_API	MC : Vvoie déclenchée sur deux automates	35													
20	X	VOR_STOCKAGE	Vvoie déclenchée sur stockage		Sort	0	K / B VOR_STOCKAGE	MC : Vvoie de stockage	21													
21	X	DEF_COMPTAIGNE	Nombre de rotations du compteur voie		Sort	0	U / B DEF_COMPTAIGNE	DEF AC : Défaut de rotations du compteur voie	31													
22	X	OPT_CE	Valeur du compteur voie		Sort	0	U / B OPT_CE	MC : Objet en redéploiement	27													
23	X	ET_DCC_DR	Objet en redéploiement		Sort	0	U / B Redéplo.	MC : Objet en redéploiement	27													
24	X	ET_BNC_DA	Objet en undeployement		Sort	0	U / B OPT_DA	MC : Objet en undeployement	27													
25	X	CMD RECADRAIS VOIE	Commande de redressement		Sort	0	O / B CMD RECADRAIS VOIE	EXO AC : Commande de redressement	30													
26	X	NEW_CPT_CE	Valeur de redressement du compteur voie		Sort	0	O / I NEW_CPT_CE	MC : Valeur de redressement du compteur voie	41													
27	X	OPT_CE	Valeur du compteur voie		Sort	0	M / I OPT_CE	MC : Valeur du compteur voie	28													
28	X	PL_011_HF_VOI	AVFER_PN18		P2_PN18	HF	P2	PAL_P2	V01													
29	X	Identifiant	Libellé		E/S	Adressé	Instruction/ Mise à l'échelle	Type	Catégorie	Prise	Prise	Prise	Prise	MIN API	MAX API	MIN physique	MAX physique	Opérations	Periode	Int (Automatique)	Libellé compact (automatique)	Taille Libellé
30	X	PL_011_HF	Mod d'état		Sort	0	I / T5 ADDRESS	MC : Mod d'état	15													
31	X	ET_BNC	Etat encodeur		Sort	0	I / B ET_BNC	INF AC : Etat encodeur	24													
32	X	ET_DCC	Etat encodeur		Sort	0	I / B ET_DCC	INF AC : Etat encodeur	21													
33	X	ET_MDC	Etat encodeur en protection		Sort	0	I / B ET_MDC	INF AC : Etat encodeur en protection	22													
34	X	ET_ADDRESS	Modèle Marconi de I / T5 ADDRESS		Modèle Marconi de I / T5 ADDRESS	0	M / I T5 ADDRESS	(MC : Valeur Marconi de I / T5 ADDRESS)	39													
35	X	ET_BNC_DR	Modèle Marconi de I / B ET_BNC		Modèle Marconi de I / B ET_BNC	0	M / B ET_BNC_DR	(MC : Valeur Marconi de I / B ET_BNC)	36													
36	X	ET_BNC_DA	Modèle Marconi de I / B ET_DCC		Modèle Marconi de I / B ET_DCC	0	M / B ET_BNC_DA	(MC : Valeur Marconi de I / B ET_DCC)	35													
37	X	ET_DCC_DR	Modèle Marconi de I / B ET_MDC		Modèle Marconi de I / B ET_MDC	0	M / B ET_DCC_DR	(MC : Valeur Marconi de I / B ET_MDC)	34													
38	X	CMD RECADRAIS VOIE	Commande de redressement		Sort	0	O / B CMD RECADRAIS VOIE	EXO AC : Commande de redressement	30													
39	X	NEW_CPT_CE	Valeur de redressement du compteur voie		Sort	0	O / I NEW_CPT_CE	MC : Valeur de redressement du compteur voie	41													
40	X	OPT_CE	Valeur du compteur voie		Sort	0	M / I OPT_CE	MC : Valeur du compteur voie	28													
41	X	ET_DCC_DA	Valeur du compteur voie		Sort	0	M / B ET_DCC_DA	MC : Valeur du compteur voie	24													
42	X	CMD RECADRAIS VOIE	Commande de redressement		Sort	0	O / B CMD RECADRAIS VOIE	EXO AC : Commande de redressement	30													
43	X	NEW_CPT_CE	Valeur de redressement du compteur voie		Sort	0	O / I NEW_CPT_CE	MC : Valeur de redressement du compteur voie	41													
44	X	OPT_DA	Valeur du compteur voie		Sort	0	M / I OPT_DA	MC : Valeur du compteur voie	28													
45	X	OPT_CE	Valeur du compteur voie		Sort	0	M / I OPT_CE	MC : Valeur du compteur voie	28													
46	X	ET_BNC_DR	Valeur du compteur voie		Sort	0	M / B ET_BNC_DR	MC : Valeur du compteur voie	28													

FIGURE 26 – Fichier de simulation.

Pour cette macro, la seule partie que je n'ai pas modifiée est la fonction GetFileNameFromPath qui permet de récupérer le nom du fichier après avoir entré en paramètre le chemin du fichier. Les trois autres fonctions sont écriture\_blocs, copier et insertion. La première permet de remplir les onglets du fichier final, la deuxième permet de formater les colonnes du fichier final et la troisième est la fonction principale de la macro.

En premier lieu, la macro choisit un nom de fichier pour le fichier final qui sera "P2\_BDD\_yyyymmdd.xlsx" avec yyymmdd qui représente la date du jour de création du fichier. Ensuite, elle copie le fichier modèle sous le nom du fichier final. Puis elle remplit les onglets SPL, SIG et VOI. Les autres onglets seront remplis plus tard avec l'avancement du projet, sauf l'onglet API que j'ai rempli à la main car il n'y a que 4 automates sur le secteur Avalfer, 1 pour chaque sous-secteur, et 1 principal.

La fonction copier prend comme paramètre le nom du fichier dans lequel coller les données. Pour cela, on demande à l'utilisateur de choisir le modèle de base de données qu'il désire copier. Après le choix du fichier, on vérifie que le fichier que l'on veut obtenir n'existe pas encore, sinon on renomme notre fichier final en ajoutant (2) ou (3),... à la fin du nom. Après avoir créé notre fichier, on l'ouvre et on met en forme les colonnes.

La fonction écriture\_blocs va remplir les onglets. Elle prend en paramètre un nom du modèle de base de données. Son exécution consiste à boucler sur chaque groupe voulu, ici les aiguilles simples, voies et signaux. Pour chaque groupe d'objets, on va grouper les objets des 3 secteurs dans un tableau avec leur nom, leur groupe et leur aire. Ensuite, un formulaire apparaît pour demander à l'utilisateur combien de lignes des onglets SPL, VOI et SIG il veut copier. Puis, pour chaque objet dans le tableau précédent, on colle les lignes copiées dans le fichier final dans le bon onglet avec le changement de nom, de groupe et d'aire de chaque objet.

Au final, on obtient un fichier nommé "P2\_BDD\_20170714 (5).xlsx" (cinquième fichier créé le 14 juillet 2017 par la macro) contenant les onglets SPL, SIG, VOI mis à jour par la macro et l'onglet API rempli à la main.

Pour tester la validité du fichier, après avoir choisi ce que l'on veut exporter dans l'onglet Aide en figure 27, on exécute une macro qui permet de créer un fichier CSV que l'on importe dans le robot. S'il y a la moindre erreur, l'import des données dans le robot aura échoué.

A	B	C	D	E	F	G	H	I	J
Note 4 :	En cas de spécifications lors du remplissage de la <b>BDD</b> , un commentaire figure, dans l'onglet concerné, sur la première case représentative. C'est principalement le cas pour l'onglet "TS" qui possède quelques subtilités.								
Description des différents onglets									
Onglets	Description	Export	Modèle						
Aide	Aide sur l'utilisation du fichier								
Param	Paramètres de la <b>BDD</b>								
Grp-Def	Groupes de défauts								
API	Automate	x	SMOD_UU_API						
AUTOR	Autorsation		SMOD_UU_AUTOR						
BANDEAU	Gère le bandeau et les <b>pop-ups</b> de commandes (sauf la création d'itinéraire)		SMOD_UU_BANDEAU						
CRITI	Gère la création d'itinéraire		SMOD_UU_CRITI						
LISTE_ITI	Liste d'itinéraire		SMOD_UU_LISTE_ITI						
MAGNETO	Magnétoscope		SMOD_UU_MAGNETO						
PROFIL	Gère les droits des utilisateurs		SMOD_UU_PROFIL						
RDP	Gestion des clients légers		SMOD_UU_RDP						
RxCx	Cartographie des automates		SMOD_UU_RxCx						
SURV	Gestion des serveurs		SMOD_UU_SURV						
TSx	Remontées des infos TOR du N1		SMOD_UU_TSx						
ES	Carte d'entrée/sortie TOR		SMOD_UU_RxCx_ES						
PNx	Passage à niveau		SMOD_UU_PNx						
VOI	Gestion des voies	x	SMOD_UU_VOI						
SIG	Gestion des signaux	x	SMOD_UU_SIG						
CEx	Compteurs d'essieux		SMOD_UU_CEx						
SPL	Aiguille Simple	x	SMOD_UU_AIG_SPL						
TJD	Aiguille trajectoire à jonction double		SMOD_UU_AIG_TJD						
TJS	Aiguille trajectoire à jonction simple		SMOD_UU_AIG_TJS						
BAR	Barrière PN		SMOD_UU_BAR						

FIGURE 27 – Fichier de simulation : onglet Aide.

#### 4.3 Prise en main de ArchestrA et des serveurs SQL

Afin que je puisse aider au mieux le développement de la supervision, l'ingénieur informaticien que j'assiste sur le projet m'a expliqué plus précisément le fonctionnement de la supervision et des serveurs utilisés. Tout d'abord, l'ingénieur informatique m'a expliqué l'organisation des serveurs. Pour le projet ULI, il y a 6 serveurs : 2 serveurs de traitement de données, 2 serveurs de communication et de création d'objet, 1 serveur SQL d'historisation et un serveur sur lequel est effectué le développement sous ArchestrA IDE, le serveur sur lequel j'ai passé la majorité de mon stage. Le doublement de certains serveurs est nécessaire car les traitements de données et les communications avec l'automate sont des points critiques du projet. Si l'un des serveurs venait à cesser de fonctionner pour la moindre raison, le second serveur prend le relais le temps que les problèmes sur le premier serveur soient résolus. Sans doublon des serveurs, au moindre problème, la supervision ne fonctionne plus et peut engendrer d'importants blocages au sein d'Arcelor-Mittal dû à l'arrêt de tous les trains de l'entreprise.

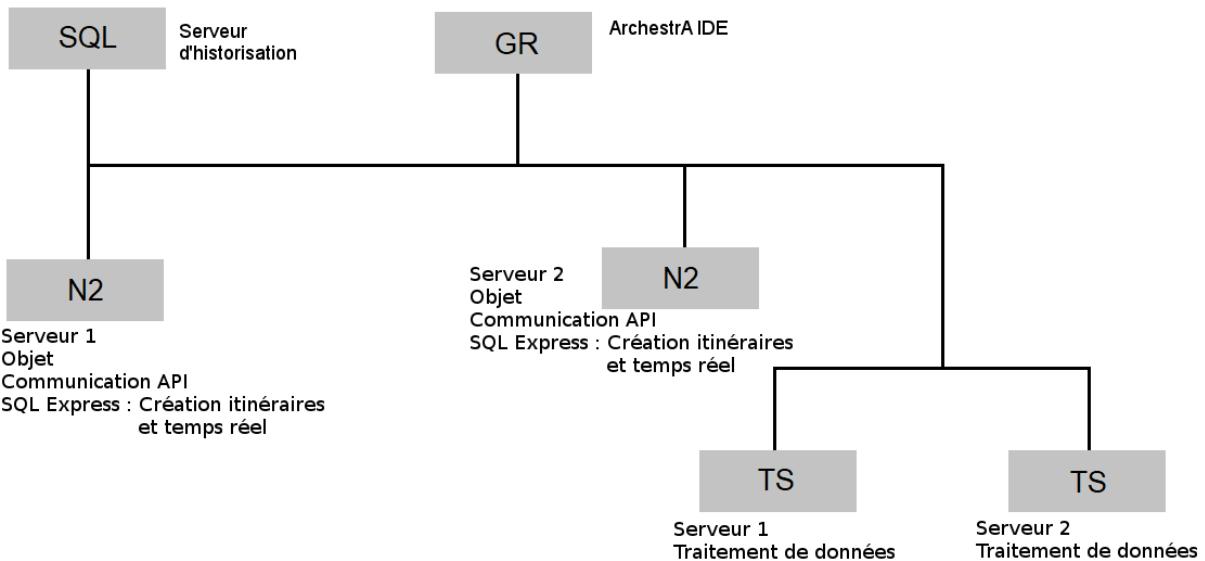


FIGURE 28 – Serveurs du secteur Avalfer.

La figure 28 ci-dessus récapitule la disposition des serveurs avec leurs fonctionnalités. La supervision nécessite les quatre types de serveurs lors de son fonctionnement. Après action de l'utilisateur sur la supervision, celle-ci va se connecter au serveur GR pour mettre à jour les valeurs des variables, puis au serveur SQL afin d'historiser chaque modification avant de se reconnecter au serveur GR afin d'envoyer les données au serveur TS qui va transmettre les données à l'API. Après exécution par le robot, on se reconnecte au serveur TS qui va transmettre au serveur GR si les modifications ont bien été effectuées, pour enfin afficher les modifications sur la supervision. La figure 29 récapitule les serveurs utilisés.

**Supervision — GR — SQL — GR — TS — API — TS — GR — Supervision**

FIGURE 29 – Serveurs utilisés lors de la supervision.

Pour se connecter à ces serveurs, je devais utiliser la connexion bureau à distance de Windows que l'on trouve en tapant *Windows + R* puis *mstsc*, puis je devais me connecter avec l'adresse IP du serveur qui est 10.7.232.121. J'arrivais alors sur une page de connexion utilisateur Windows où je me connectais en tant qu'administrateur du serveur. À partir de là, je pouvais accéder à ArchestrA.

La création d'itinéraire ne se fait pas dans ArchestrA, mais dans SQLServer, dont le langage est légèrement différent du MySQL. À titre d'exemple, le code en figure 30 ci-dessous, en MySQL donnerait

```
SELECT Index, VOIE_ORIG, VOIE_DEST, SIGNAL, SENS, COMMANDE
FROM FA_ITINERAIRE.dbo.Liste_Itineraire
LIMIT 1000
```

	Index	VOIE_ORIG	VOIE_DEST	SIGNAL	SENS	COMMANDE
1	1	26	28	154	1	JBZ&Jftft
2	2	28	26	159	2	-JBZ&Jftft
3	3	25	26	155	2	-J ftft
4	4	26	25	154	1	J ftft
5	5	28	36	161	1	\$JBZ&Jftft
6	6	36	28	163	2	JBZ&Jftft
7	7	31	27	151	2	-J  ftft
8	8	27	31	152	1	J ftft
9	9	31	32	151	2	JB  ftft
10	10	32	31	153	1	JB4ftft
11	11	27	28	162	2	JB8ftft

FIGURE 30 – Code SQLServer de sélection d'itinéraires.

Afin de saisir les quelques nuances, j'ai été chargé de modifier la procédure SearchITIAuto qui permet de rassembler tous les itinéraires possibles dans une même table. Avant tout, il faut savoir que les commandes sont uniques entre 2 voies consécutives, séparées par une seule aiguille, contiennent 4 caractères spéciaux et se terminent par "ftft". Si 2 voies non consécutives peuvent être reliées, la commande reçue par l'automate sera la concaténation des 2 commandes sans le premier "ftft". Par exemple, si les voies 1 et 3 peuvent être reliées en passant via la voie 2 et que la commande entre les 1 et 2 est  $\exists \leftarrow \infty \theta$  ftft et que celle entre les voies 2 et 3 est  $\mu \xi \leftarrow \cong$  ftft, alors la commande pour l'itinéraire entre les voies 1 et 3 sera  $\exists \leftarrow \infty \theta \mu \xi \leftarrow \cong$  ftft. Afin que je comprenne cette partie, j'ai modifié la procédure en y intégrant cette dernière particularité. Ensuite, j'ai travaillé sur ArchestrA afin de pouvoir créer les vues du projet. J'ai tout d'abord commencé à comprendre comment fonctionnent les Galaxies, les objets et tout ce qui en découle. J'ai commencé par me pencher sur les instances d'objet. Lorsque que l'on choisit de modifier un objet ou une de ses instances, on tombe sur une fenêtre de variables comme en figure 31. Sur cette fenêtre, il y a deux parties : la partie avec toutes les variables au centre de la figure

31, et la partie avec les détails de la variable choisie, tels que le nom, la description, le type, la valeur initiale, ...

Pour le projet, le nom des variables a été normé par le client, il doit être composé de trois parties. La première est la provenance de la variable, soit U si c'est une variable saisie par l'utilisateur, I si c'est une variable d'entrée, IO si c'est une variable d'entrée-sortie, K si c'est une constante, M si c'est une variable magnétoscope. La deuxième est le type de variable, I pour un entier, B pour un booléen, S pour une chaîne de caractère. La troisième est le complément du nom et est laissée au choix du développeur.

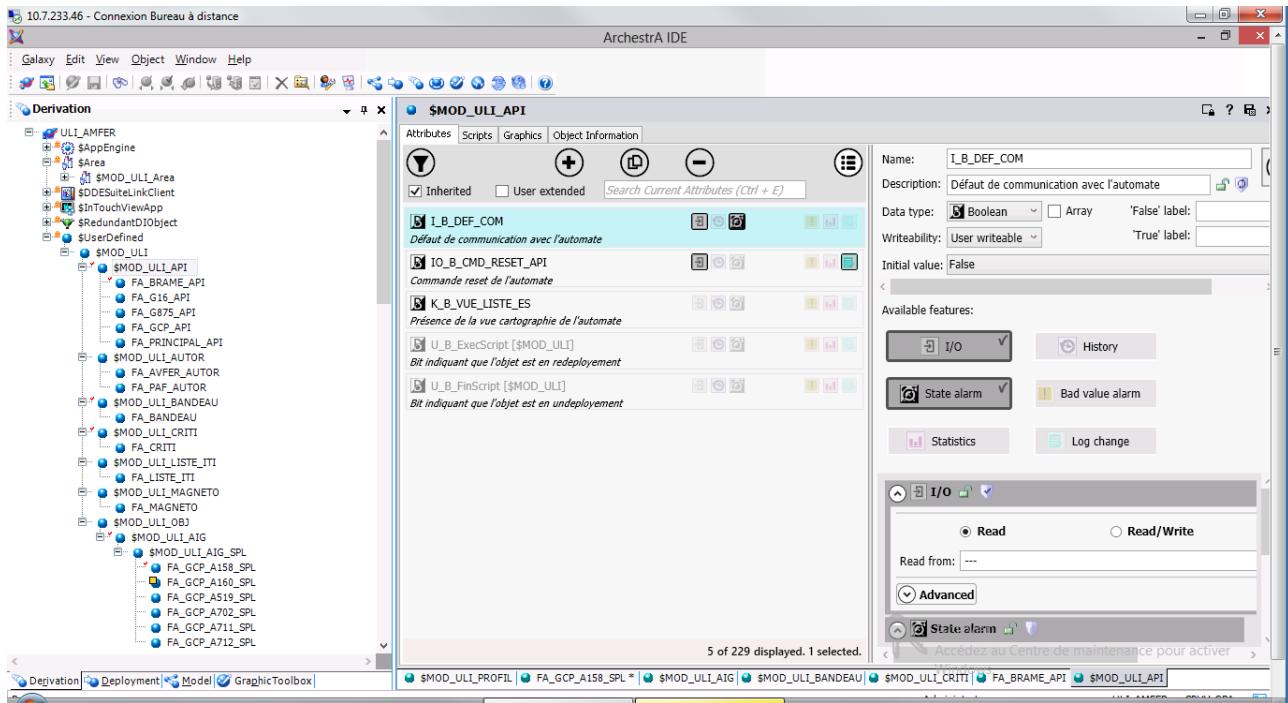


FIGURE 31 – Fenêtre de variables sous Archestra.

La description est nécessaire pour expliquer brièvement à quoi sert la variable. Les champs “Data type” et “writeability” doivent correspondre avec la norme du nom de la variable. Les caractéristiques disponibles sont optionnelles et j'en ai rencontré trois lors du projet qui sont “I/O”, “History” et “State Alarm”. La première permet de définir une variable d'entrée-sortie, puis de sélectionner si c'est une variable d'écriture ou de lecture-écriture et son chemin d'accès. La deuxième permet de définir la variable comme variable magnéto. La troisième définit la variable comme variable d'alarme se déclenchant dans certains cas.

Après la partie variable, je me suis penché sur la partie script d'un objet ou d'une instance qui est le deuxième onglet de la fenêtre de l'objet. Cette partie répertorie tous les scripts disponibles pour cet objet ou cette instance et permet d'en créer. Les scripts peuvent être programmés pour se lancer au changement de valeur d'une variable ou lors d'une condition. Ce sont en général des scripts qui ne changent pas l'état de l'objet sur la vue où ce dernier est utilisé. Pour ces scripts-ci, ils sont généralement employés sur les graphiques de l'objet, qui sont le troisième onglet de la fenêtre .

La partie graphique de l'objet est l'endroit où tout les objets graphiques qui font partie de l'objet sont mis. Par exemple, pour l'objet VOI, les objets graphiques nécessaires étaient la voie et le nom de la voie. Les objets graphiques de l'objet ou l'instance sont d'abord créés dans le menu “Graphics” d'Archestra avant d'être importés sur l'éditeur graphique de l'objet, en figure 32. Le quatrième onglet regroupe les informations de l'objet ou de l'instance.

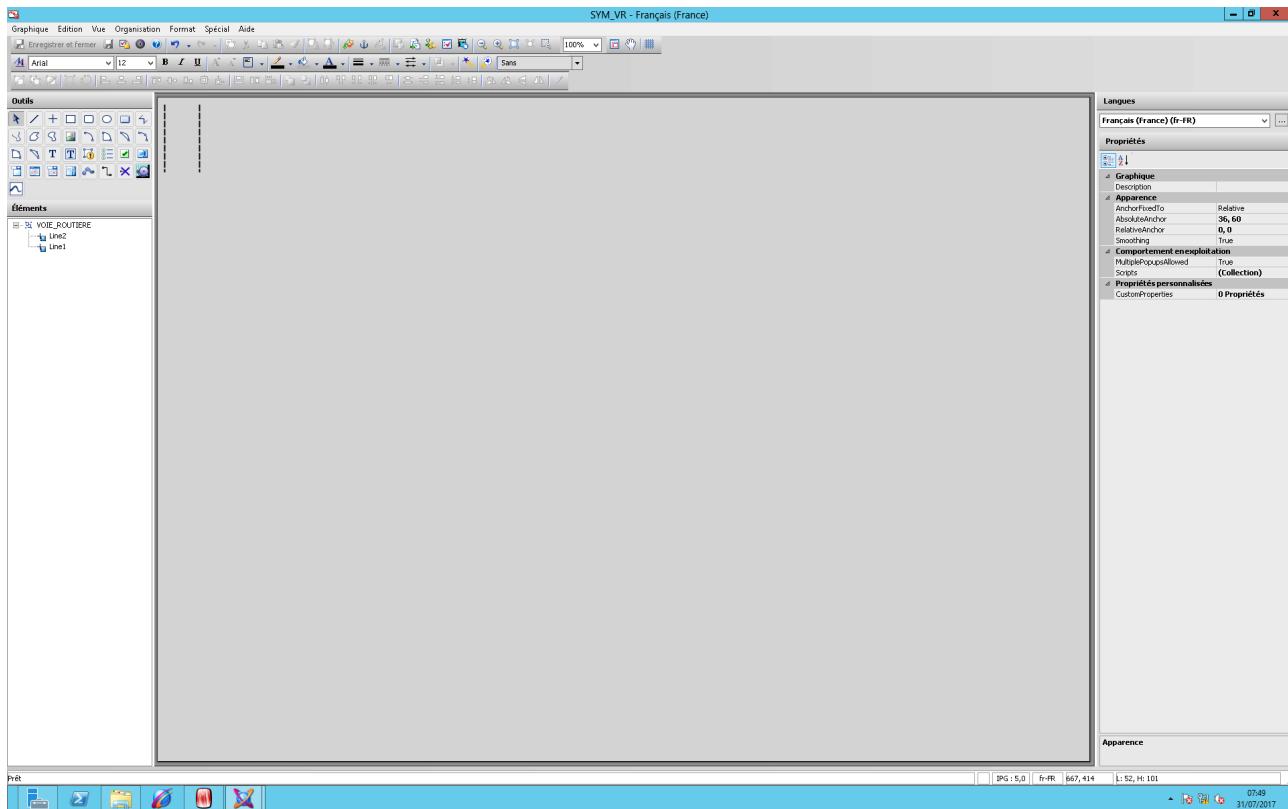


FIGURE 32 – Éditeur graphique de l’objet PN contenant l’objet graphique VOIE\_ROUTIERE.

#### 4.4 Désactivation de clavier et de souris d’une Raspberry Pi 3

Cahier des charges : *La RaspBerry doit pouvoir accepter tout périphérique USB branché avant son lancement, mais interdire ceux branchés après son lancement. Un périphérique autorisé qui est débranché doit être autorisé et fonctionnel. Elle doit être prête pour le jeudi 27 juillet.*

En travail secondaire, en parallèle au projet ULI, on m'a demandé d'améliorer la sécurité de la Raspberry. L'objectif était de pouvoir connecter une clé USB à la Raspberry afin que celle-ci copie un diaporama, éjecte la clé et lance le diaporama à son démarrage. Et je devais permettre à la Raspberry de refuser tout périphérique USB branché après son démarrage hormis ceux connectés au démarrage et débranchés lors de l'utilisation de la Raspberry. N'ayant jamais travaillé sur une Raspberry et que très peu sous Linux, la tâche fut relativement compliquée. J'ai heureusement trouvé une explication sur Internet qui est proche de ce que je voulais faire. Cependant, après test, la Raspberry acceptait encore tous les périphériques USB. En essayant de trouver pourquoi cela ne fonctionnait pas, j'ai étudié les règles Udev. Ce sont des règles qui gèrent les périphériques USB. Ainsi, j'ai pu comprendre le fonctionnement de ce qui était expliqué sur le site. Mais par manque de temps, je n'ai pas pu trouver pourquoi cela ne fonctionnait pas.

## 4.5 Crédation de vues pour le projet ULI

Le plus gros travail de mon stage a été la création de vues pour le projet ULI. J'ai été chargé de créer la vue du secteur, ainsi qu'une fenêtre pop-up et un menu des travaux.

**Vue principale** Cahier des charges : *En respectant l'exemple donné par Arcelor-Mittal, il faut créer la vue du secteur "Amonfer". Pour cela, il faut placer chaque objet graphique sur un nouvel objet graphique et les aligner afin d'obtenir la plus fidèle représentation possible. Elle doit être fonctionnelle le plus rapidement possible.*

Pour créer cette vue, j'ai dû créer un objet InTouchViewApp \$ULI\_AVFER dont l'instance ULI\_AVFER\_P2 est la vue InTouch qui sera affichée au client. C'est cet objet qui permet à ArchestrA et InTouch de communiquer. Une fois l'instance créée, je devais créer un dossier "Vue" avec la vue dans l'onglet "Graphic Toolbox" d'ArchestrA en figure 33. C'est l'objet graphique G16 sur la figure 33 qui sera modifié pour obtenir la vue finale.



FIGURE 33 – Onglet “Graphic Toolbox” d'ArchestrA.

La vue doit avoir une taille de 1920X1080. Au commencement, je pensais pouvoir séparer la vue en trois sous-objets graphiques, un par sous-secteur (G16, P3, TRI) puis les assembler sur une même vue. Cependant, après avoir compris comment faire et fait les trois parties, on m'a demandé de tout mettre sur une seule vue directement. Je devais donc redimensionner les objets afin que la vue finale corresponde le plus fidèlement possible au modèle donné par le client. Lorsque j'eus terminé la vue, l'ingénieur informatique m'a fait remarquer que les objets que j'avais importés n'étaient pas les objets graphiques des instances d'objet, mais les objets graphiques génériques de l'onglet "Graphic Toolbox" d'ArchestrA. Lors du lancement de la supervision, la vue n'aurait pas été animée. J'ai donc dû tout recommencer en important les bons objets graphiques. Ce qui fait environ 200 objets graphiques à importer, redimensionner et placer. Comme la vue sera montrée telle qu'elle au client, tous les objets devaient être parfaitement alignés, au pixel près. L'alignement sur l'éditeur graphique d'ArchestrA s'est fait assez rapidement malgré le fait que les objets redimensionnés qui étaient formés de plusieurs sous-objets graphiques se désassemblaient si la réduction ou l'agrandissement ne respectait pas le ratio de base. Lorsque j'ai importé la vue sur InTouch, tous les objets ont été désorganisés et le seul moyen de les réaligner était de les modifier sur la vue ArchestrA pour que cela se répercute

sur InTouch. Évidemment, en lançant le Viewer, les objets étaient également désorganisés par rapport à la vue InTouch. Le déplacement de la vue d'un pixel sous InTouch permettait de réaligner certains objets et d'en déranger d'autres. L'objectif final étant une vue parfaite sur le Viewer, l'alignement s'est fait au pixel près, objet par objet sur ArchestrA pendant environ 2 semaines, en essayant de modifier les objets graphiques et les mettre directement à la taille voulue sur leur éditeur graphique d'ArchestrA, avant que l'on me propose une autre création d'objet graphique, la pop-up des horaires des PN.

**Pop-up des horaires des PN.** Cahier des charges : *La fenêtre pop-up doit être créée sur une nouvelle fenêtre InTouch. Elle doit apparaître au moment voulu par le client, soit 7h40 et 16h20, et ne doit disparaître qu'au clic sur le bouton OK. Elle doit être opérationnelle au plus tôt.*

Un visuel de la fenêtre est donné en fig34.



FIGURE 34 – Pop-up des horaires des PN.

Le but de la pop-up est d'apparaître à deux horaires donnés : 7h40 et 16h20, et de disparaître que lors d'un clic sur le bouton "OK". Pour cette image en figure 34, j'ai créé un dossier "Pop-up" dans la "Graphic Toolbox" et une instance d'InTouchViewApp "essai". Tous les éléments de la fenêtre, excepté le bouton, ont été créés à la main. Le bouton sert à faire disparaître la fenêtre. J'ai juste utilisé un changement de variable de visibilité passant à faux lors du clic sur le bouton "OK". Sur InTouch, j'ai géré l'apparition de l'objet avec deux variables globales internes à InTouch "\$Hour" et "\$Minute" qui récupèrent l'heure et les minutes du serveur, puis j'ai utilisé un script "Condition" et un "Data Change" qui affichent l'objet de la fenêtre aux heures voulues. Ainsi, j'ai pu utiliser les scripts "Condition" et "Data Change".

**Modification de l'objet PN et VOI** Cahier des charges : *L'objet PN contient des attributs qui appartiennent à un objet barrière. En respectant l'objet PN de base et ses variables déjà existantes, il faut créer un objet barrière afin que ce dernier soit indépendant. Il faudra également faire en sorte que l'objet PN ait une variable avec le nombre de barrières qui lui appartiennent. Les scripts présents sur l'objet PN devront également être modifiés afin de garder la condition suivante : passage de la voie routière en vert si toutes les barrières sont ouvertes. À faire au plus vite.*

Pour consolider les bases que j'ai acquises, on m'a demandé de modifier l'objet PN car il ne correspondait pas aux attentes du client lors d'une réunion de projet. L'objet PN contenait à l'origine les objets barrières, mais pour des soucis d'optimisation, je devais sortir les objets barrières de l'objet PN, créer un objet BAR et les objets graphiques qui en découlent. Sur le terrain, chaque PN dispose de 3 zones : 2 zones d'attentes (ZA) et une zone de maintien (ZM). Chaque zone dispose de 2 capteurs et d'un certain nombre de barrières. J'ai donc remplacé les variables et les scripts des barrières, stockés dans l'objet BAR (graphique en figure 35), par les variables et les scripts des capteurs. L'objectif final étant que le PN passe au vert lorsque

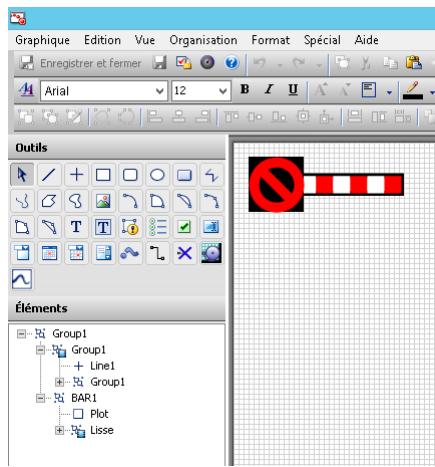


FIGURE 35 – Objet BAR.

les barrières de celui-ci sont toutes ouvertes. La variable du PN, nommée “U\_I\_BAR\_OUV” est vraie seulement si les variables correspondantes des objets BAR sont également vraies. Un exemple de PN fonctionnel sera donné sur la vue finale de mon projet.

Sur le modèle donné par le client, les noms de voies sont représentés avec un rectangle simple, juxtaposés avec une flèche et/ou un rectangle à gauche et/ou à droite. Je devais modifier l’objet graphique “NOM\_VOI” afin de satisfaire les conditions. Je devais alors ajouter un rectangle et une condition de visibilité avec une variable ArchesTrA qui affiche le nom sous un rectangle simple si elle vaut 0, avec les deux flèches si elle vaut 1, avec une flèche et un rectangle si elle vaut 2, avec une simple flèche si elle vaut 3. Je n’ai pas utilisé plus de conditions car la modification miroir sur l’objet ne retourne pas le texte, donc les conditions flèche à droite seule ou flèche à gauche seule sont les mêmes conditions.

Après la création ou modification d’objet, ces derniers sont présents que sous ArchesTrA, en local. Pour qu’ils puissent être utilisés et apparaître sur la supervision, il faut tout d’abord que l’objet ne contienne aucune erreur (variable inconnue après placement dans une aire) et qu’il soit “Checked-in” puis déployé après avoir modifié. Sans ces étapes, l’objet ne peut pas être utilisé sur la supervision. Chaque erreur énumérée peu avant est représentée différemment sur ArchesTrA. L’erreur est représentée par un carré rouge avec une croix blanche, un objet “Checked-Out” est indiqué par une coche rouge à gauche de son nom, un objet non déployé possède un carré jaune et un objet modifié mais non undeployed possède un carré jaune et noir, comme montré en figure 36.

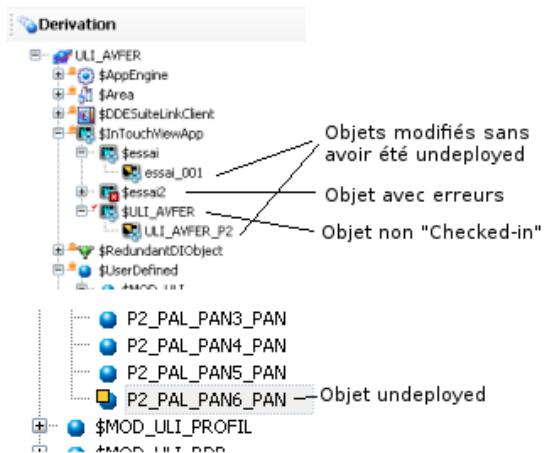


FIGURE 36 – Objets inutilisables par la supervision.

**Menu des travaux** Cahier des charges : *Cet objet graphique doit permettre l'affichage de panneaux de travaux sur la vue générale. Un panneau doit pouvoir être agrandit, rétréci par des flèches présentes sous le panneau et doit contenir une zone de texte pour y disposer des infos ainsi qu'un bouton "Fin" pour le réinitialiser. Le menu doit contenir cinq panneaux ainsi qu'un bouton de réinitialisation de tous les panneaux et un bouton pour cacher le menu. Le menu doit pouvoir apparaître et disparaître avec les panneaux qui n'ont pas été placés sur la vue, lors d'un clic sur la zone de texte. Le menu doit pouvoir être déplacé et caché ou découvert sur une session indépendamment d'une autre section. Le menu doit pouvoir se trouver dans le coin haut à droite sur un supervision mais au centre sur une autre et dans le coin inférieur droit sur une troisième sans que cela ne dérange le déplacement des panneaux.*

Les visuels des objets graphiques sont donnés en figure 38 pour le panneau de travaux et en figure 39 pour le menu.

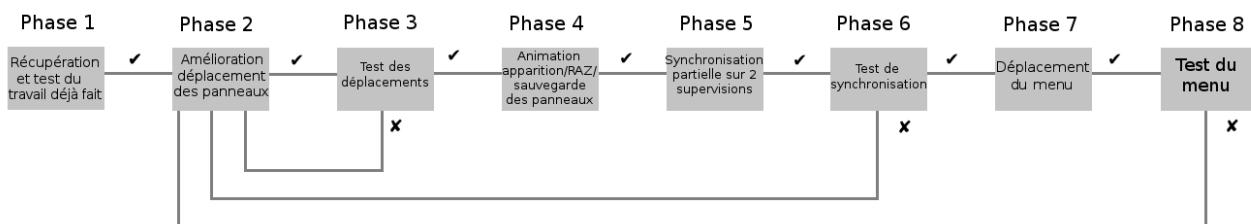


FIGURE 37 – Schéma de conception du menu des travaux.

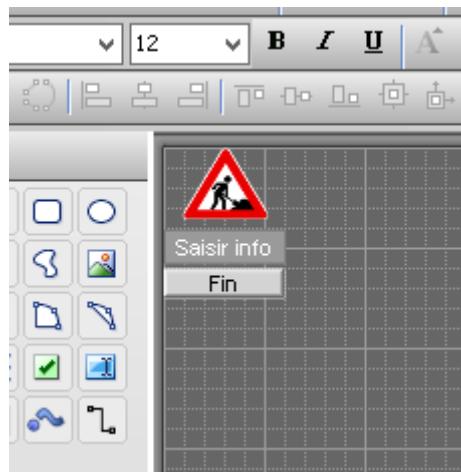


FIGURE 38 – Panneau d'avertissement des travaux.



FIGURE 39 – Menu des travaux.

Le menu est à intégrer à la vue générale, donc il n'y avait pas d'instance d'InTouch à créer. Ce menu avait été commencé par un collaborateur d'Arcelor-Mittal, mais sans avoir été testé. J'ai donc reçu les objets graphiques et je devais assembler le tout afin de faire un menu mobile grâce auquel on peut déplacer des panneaux d'avertissement de travaux sur les voies. Phases 1 et 2 du schéma de conception. En premier lieu, je devais créer l'objet afin qu'il soit fixe et que seuls les panneaux puissent bouger et s'agrandir ou rétrécir ; au clic sur l'image du panneau, certaines informations devaient apparaître ou disparaître comme les flèches de changement de taille (cachées derrière le panneau), les infos et le bouton de Fin. Pour cela, j'ai créé 5 instances de l'objet “\$MOD\_ULI\_PAN” et j'ai utilisé une animation “Curseur horizontal” et “Curseur vertical” qui permettent de déplacer le panneau. Sur la figure 40, la référence est la variable du curseur, les positions gauche et droite sont les valeurs minimale et maximale de la référence et le mouvement vers la gauche ou la droite est le nombre de pixels dont on peut déplacer l'objet, l'ancrage est là où se placera la souris pour déplacer l'objet et l'écriture des données est le moment où se fait le stockage de la valeur dans la référence. J'ai pris un mouvement vers la droite et vers la gauche de 1980px car c'est la largeur de l'écran. Un mouvement variable selon la position de référence de l'objet dans le menu fonctionnait, mais posait des problèmes pour la dernière partie du menu.

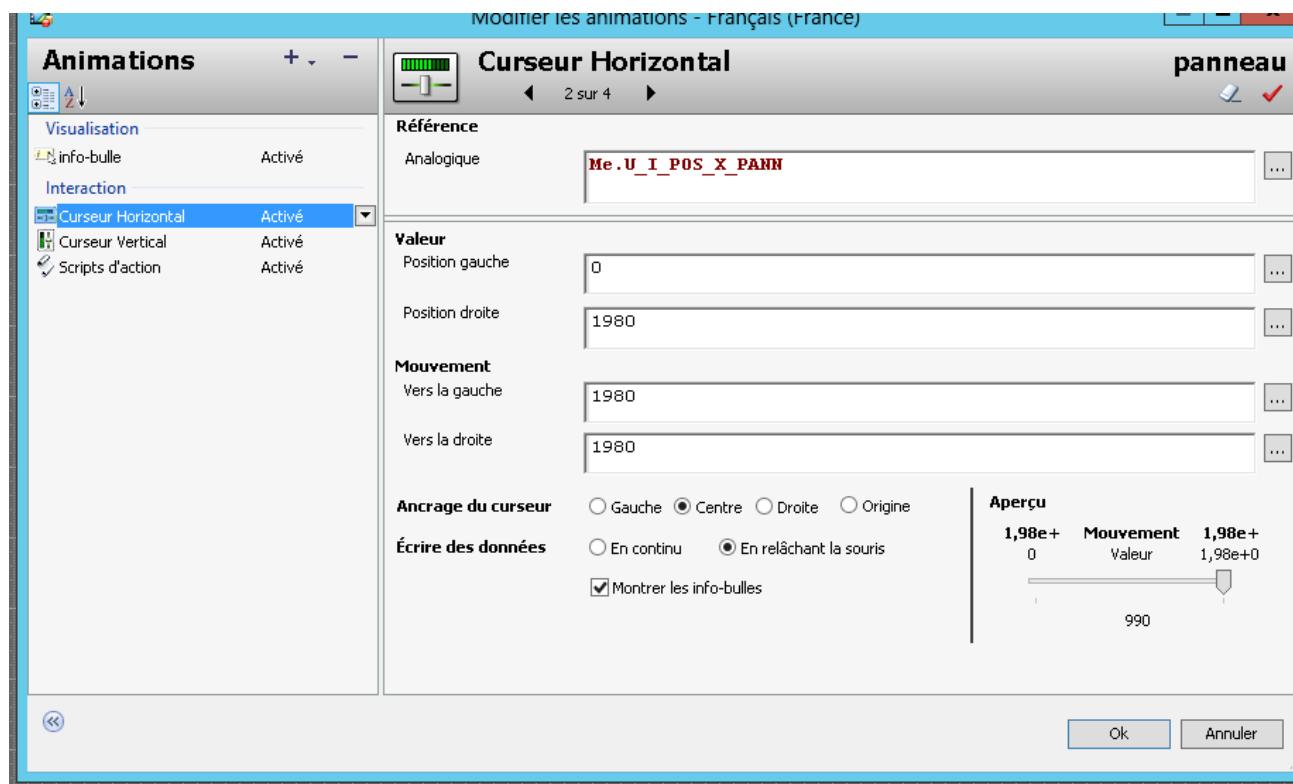


FIGURE 40 – Animation curseur sous ArchestrA.

En deuxième lieu, phases 4 à 6, je devais permettre l'affichage complet ou non du menu ainsi que la remise à zéro de tous les panneaux et la sauvegarde des positions des panneaux déplacés pour qu'au lancement du Viewer, les panneaux n'aient pas changé de place. De plus, il fallait que je gère une certaine synchronisation entre deux affichages simultanés de la vue, lors d'une connexion administrateur et invité, par exemple.

Pour l'affichage, j'ai utilisé une animation de visibilité sur tout les éléments contenant dans le plus rectangle de la figure 39. Lors d'un clic sur le texte “Réduire / Agrandir”, les éléments devenaient visibles ou invisibles et lors d'un clic sur la croix, les éléments se cachaient. J'ai donc utilisé un booléen pour la visibilité qui passaient à son inverse lors du clic sur le texte et à false lors d'un clic sur la croix. Les panneaux déjà déplacés devaient rester visibles tant qu'ils

n'avaient pas été réinitialisés.

Pour la remise à zéro des panneaux, j'ai commencé par m'occuper de la remise à zéro d'un seul panneau. Pour cela, j'ai dû utiliser un élément graphique permettant de repositionner le panneau sur celui-ci. J'ai donc utilisé un rectangle de mêmes dimensions que le panneau et lors de la remise à zéro, je forçais les coordonnées du panneau à valoir les coordonnées du rectangle créé peu avant, ainsi que les valeurs du curseur qui reprenaient leur valeur de départ. J'ai également forcé les dimensions du panneau et ses informations à revenir à leur état d'origine, comme montré dans le script en figure 41.

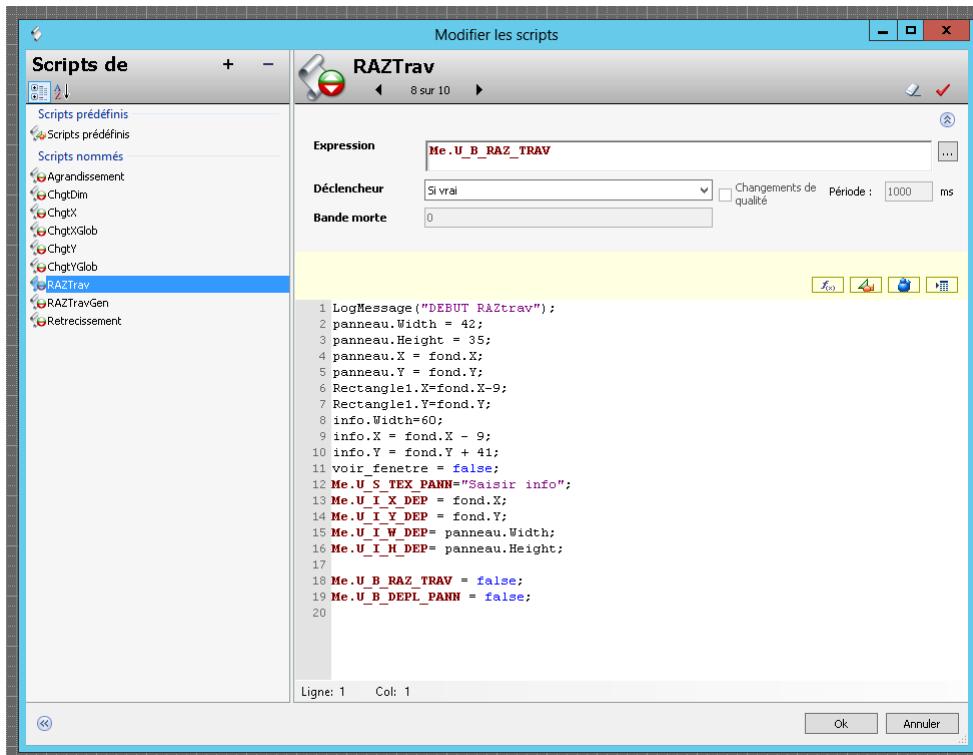


FIGURE 41 – Script de remise à zéro du panneau.

Pour la remise à zéro généralisée, j'ai utilisé une variable InTouch qui fait passer toutes les variables de remises à zéro des panneaux à true.

Pour la sauvegarde des postions et de la taille des panneau, j'ai utilisé 4 variables ArchestrA pour sauvegarder à chaque fermeture du Viewer les positions des curseurs et les dimensions du panneau grâce à un script "Prédéfini à la fermeture", et qui actualise les variables lors d'un script "Prédéfinis à l'ouverture". Un exemple de script est donné en figure 42.

Pour la synchronisation partielle de la supervision, je devais faire en sorte que les panneaux bougent et puissent être modifiés en simultanés sur les 2 supervisions, mais leur visibilité dépend si l'utilisateur cache ou non les éléments. Il pouvait donc y avoir un menu ouvert sur une supervision et un menu caché sur l'autre. Pour cela, j'ai donc dû jongler entre les variables de l'instance de l'objet sous ArchestrA et les variables InTouch qui sont des variables globales à la supervision, et les variables locales à l'objet graphique, les "Propriétés personnalisées". Il venait facilement que les curseurs du panneau doivent être des variables ArchestrA, mais la visibilité et les positions X et Y d'origine d'un panneau et de ses options devaient être des "Propriétés Personnalisées". Ainsi, je n'ai utilisé que 3 "Propriétés Personnalisées" pour le panneau, deux pour les positions du rectangle qui sert à remplacer le panneau et une pour la visibilité du panneau s'il n'est pas déplacé. Les autres variables sont des variables ArchestrA. Pour le menu, j'ai également eu besoin de "Propriétés Personnalisées" pour son affichage complet car il se fait de façon indépendante sur chaque supervision.

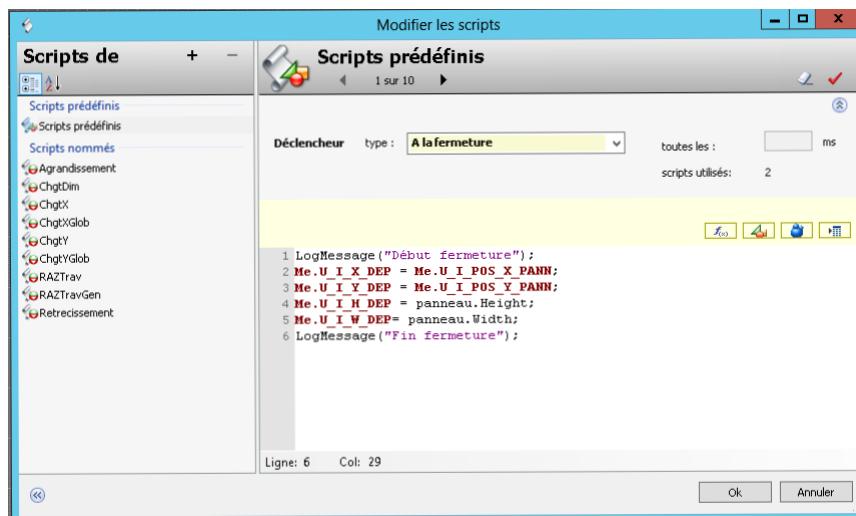


FIGURE 42 – Script prédefinis à la fermeture d'un panneau.

En dernier lieu, phases 7 et 8, je devais permettre au menu d'être déplacé sans bouger les panneaux déjà placés sur la vue. Cette partie fût la plus compliquée à mettre en place. Pour le déplacement du menu, je suis parti comme pour le déplacement d'un panneau mais avec des "Propriétés personnalisées" car le déplacement ne se fait pas sur toutes les supervisions. Deux animations curseurs, puis déplacement des éléments graphiques du menu suivants les déplacements du curseur, donc déplacements des cinq objets panneaux importés. Cependant, le mouvement des objets panneaux importés ne faisait pas bouger les coordonnées de l'élément panneau et les valeurs que j'utilisais dans les curseurs du panneau. Ce qui faisait que les panneaux se déplaçaient de manière aléatoire ou ne pouvaient se déplacer que dans une zone qui n'était pas celle désirée. C'est en changeant par hasard la valeur des curseurs que j'ai trouvé que les panneaux se déplaçaient dans la zone voulue. Un exemple en phase de test est en figure 43 où la zone de déplacement voulue est en vert et celle du panneau est en jaune.

Après quelques réglages sur les valeurs des curseurs et après suppression des scripts et variables inutiles et optimisation des scripts, le menu est opérationnel et fonctionne comme voulu sur la supervision.

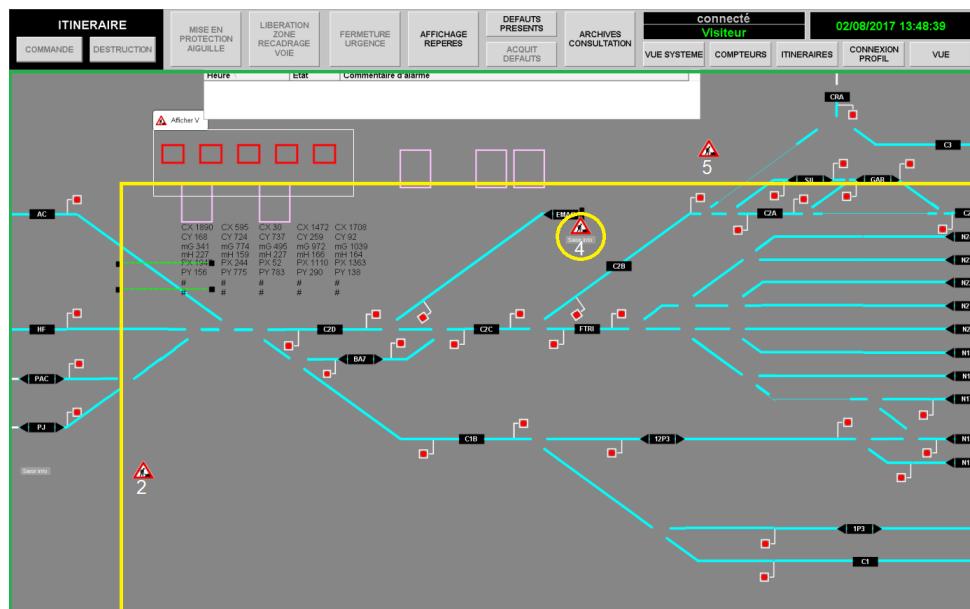


FIGURE 43 – Phase de test du menu.

## 5 Compte-rendu des activités

Chaque activité que j'ai réalisée m'a permis de voir plus précisément en quoi consiste l'informatique industrielle, comment est la vie dans un open-space et surtout ce qu'est le travail dans le développement informatique. La majorité des activités qui m'ont été confiées ont été entièrement terminées dans les délais fixés.

Ma toute première activité, les exercices de prise en main d'InTouch, n'avait pas de deadline, je devais progresser à mon rythme et surtout comprendre ce que je faisais et pourquoi je le faisais. Ces exercices m'ont donc permis de prendre en main InTouch pour être plus efficace sur le développement de la supervision qui a suivi et j'ai ainsi pu m'ouvrir à mes collègues en allant leur demander de l'aide quand je ne trouvais pas mon bonheur dans le manuel d'utilisation d'InTouch. Cette activité a été faite en une semaine.

Ma deuxième activité, la création de macros Excel, m'a permis de découvrir le langage VBA et une nouvelle facette d'Excel, logiciel que je n'utilisais jamais de cette façon. J'ai également eu la possibilité d'en apprendre un peu plus sur la partie automatisation du projet et surtout de comprendre l'utilité des fichiers Excel pour la supervision. La première macro que j'ai écrit m'a pris 3 jours sur les 5 accordés, entre le temps de comprendre ce qu'est une macro, comment l'écrire, l'écrire et tester. La seconde macro, le fichier de simulation, m'a été donnée pour le début de ma quatrième semaine de stage. Cela m'a permis de mieux comprendre comment sont utilisés les fichiers Excel. J'ai réalisé la macro en six jours, sans deadline fixée car je l'utilisais pour la suite de mes activités.

Ma troisième activité, la prise en main d'ArchestrA, m'a pris 13 jours avant d'être briefé entièrement sur la supervision du projet. Je devais être prêt en 15 jours. J'ai ainsi pu prendre en main ArchestrA et comprendre comment sont faites les supervisions.

Ma quatrième activité, la Raspberry Pi 3, a été l'activité la moins aboutie, voire même pas aboutie du tout à la deadline. Gérer les périphériques semblait à première vue une chose simple avec l'explication trouvée sur Internet, mais les choses se sont compliquées assez vite. Malgré un suivi rigoureux de l'explication, soit aucun périphérique n'était refusé, soit un périphérique branché au lancement n'était plus détecté si on le rebranchait. De plus, par une erreur d'inattention, j'avais bloqué tous les périphériques, sauf une souris. Et une souris n'est pas le périphérique le plus adapté pour entrer des lignes de commandes dans le terminal, sans clavier interne installé sur la Raspberry évidemment... J'ai donc dû copier-coller lettre par lettre chaque ligne de commande afin de débloquer la Raspberry et pouvoir rebrancher un clavier, car le fichier à modifier n'était lisible et modifiable qu'avec les droits admin. J'avais 3 semaines pour faire ce travail mais n'ayant pas réussi à trouver pourquoi les modifications ne fonctionnaient pas et personne ne pouvant m'aider à ce niveau, la Pi a été rendue sans les fichiers modifiés afin d'éviter des blocages impromptus.

Ma dernière activité, la création des vues, a été celle qui a occupée la majorité de mon stage. Entre la modification de l'objet PN et la création des objets BAR, la création de la fenêtre pop-up des horaires des PN et le menu de travaux et la création complète de la vue générale, j'ai mis 25 jours pour terminer toutes ces activités. La vue générale devait être bien avancée sous 10 jours, et avec les erreurs dans les imports d'objets graphiques et l'alignement légèrement hasardeux lors de l'import de la vue sur le Maker et lors du lancement du Viewer, la vue était terminée mais pas complètement alignée. L'ingénieur que j'assistais m'avait prévenu que l'alignement était un travail laborieux et pas fort intéressant car sur la supervision "Amonfer", l'alignement des objets lui avait pris 15 jours de travail. La vue a encore été améliorée au fil des jours suivants, mais l'alignement n'a pas été terminé à 100% à la fin de mon stage.

La modification des objets m'a permis de mieux comprendre certains détails du projet que j'avais survolés lors de la prise en main d'ArchestrA et m'a montré que le cahier des charges peut changer très rapidement lors des réunions de projet avec le client. De même avec la création de la pop-up qui est survenue après un mail du client. Mais tout cela m'a permis d'accroître

mes connaissances sur ArchestrA. Le menu des travaux semblait une activité assez facile, mais certaines animations peuvent être compliquées à utiliser, dont le curseur qui m'a posé quelques problèmes, surtout sur le fait que le changement de valeur du curseur ne peut pas se faire en rentrant la valeur à la main mais uniquement en déplaçant l'objet à la souris. De même pour les valeurs des mouvements du panneau qui ne peuvent pas être changés lors du déplacement du menu. Tous ces soucis ne peuvent pas être anticipés et sont découverts que lors des phases de test. Je pensais finir le menu en 5 jours, j'ai réussi à la finir juste avant la fin de mon stage, soit en 15 jours. Ce qui montre qu'un problème pas forcément évident à première vue peut tripler le temps passé sur une activité et qu'un projet qui semble relativement facile ne le sera pas forcément. Grâce à ça, j'ai réussi à m'adapter au logiciel ArchestrA, ce qui peut être un plus si je travaille dans une entreprise qui utilise ce logiciel.

## 6 Réflexions personnelles

Ce stage m'a permis de réfléchir sur certains points. Le premier point m'est apparu assez rapidement. Comme pour mon stage précédent, l'entreprise comporte "deux mondes" différents. Celui de ceux qui vont travailler sur le terrain et celui de l'administratif. Ceci s'est confirmé lors des repas du midi. Dans la réfectoire, il y avait toujours au moins la moitié de ceux qui travaillent sur la Plateforme de Développement, soit 4 à 5 personnes et 2-3 travaillant au Bureau d'Études, mais jamais aucun administratif durant mes 9 semaines de stage, et très peu de personnes mangent en travaillant. Au niveau de la disposition des étages, il en est de même. La Plateforme de développement est un open-space avec seulement des barrières transparentes pour délimiter les zones des projets, le Bureau d'Études est également un open-space où les chefs de projet sont également séparés par des barrières transparentes tandis qu'au second étage, les bureaux sont individualisés et soit vides, soit fermés. Il y a même un midi où nous nous sommes retrouvés uniquement à 11 développeurs de la Plateforme, soit la totalité de ceux y travaillant ce jour-là.

Le deuxième point est au niveau du relationnel client-développeur. Lors de mes 9 semaines de stage, je croisais régulièrement des clients venant de différentes entreprises, mais il y avait également des déplacements des développeurs chez le client, ce qui est propre à du travail dans le domaine industriel et ce que je n'imaginais pas du tout à mon arrivée. Mais est-ce que dans le domaine du développement Web, mobile ou logiciel, il y a également cette relation développeur-client ou est-ce que c'est plus proche du cliché du développeur qui reste toute sa journée de travail devant son écran sans voir personne à part son chef de projet et ses collègues développeurs ? Ce serait intéressant de travailler dans ces domaines afin d'avoir une idée de ce qui en est réellement.

La dernière remarque que je me suis faite est au niveau de la quantité de travail qu'ont les développeurs de la Plateforme. La plupart des développeurs travaillent en simultané sur 2 gros projets, ce qui ralentit la progression des deux projets. S'il doit arriver au développeur de se déplacer pour aller corriger un problème, cela lui fait perdre énormément de temps, surtout si l'entreprise est éloignée. Certains employés pouvaient se déplacer à Metz, Amiens, Bruxelles, et le déplacement peut durer plusieurs jours. Cette surcharge de travail m'a fait remarquer qu'il manque une chose indispensable à l'entreprise pour être plus performante : des développeurs. Et cela reflète ce qui se passe actuellement en France, il manque énormément de développeurs informaticiens dans tous les domaines. Ce qui est bénéfique pour tous ceux sortants d'une école d'informatique qui trouveront facilement du travail.

## III Synthèse

### 1 Bilan du projet

Au cours de mon stage, j'ai pu terminer la majorité de mon travail sur le projet, à savoir la modification des objets ArchestrA PN et BAR, la création de la pop-up, du menu des travaux et la vue générale. Après des tests, mon travail a été validé en théorie. En pratique, mon travail sera utilisé lors des recettes et sera donc entièrement validé ou non. Quand je suis retourné à l'entreprise le 29 août, les développeurs travaillaient sur les vues que j'avais développées sans avoir rencontré d'erreurs. La prochaine étape du projet est la phase de recettes qui doit se dérouler d'ici la fin de l'année.

### 2 Bilan personnel

Avant de commencer ce stage, je ne connaissais pas les domaines de l'automatisme et la supervision. J'ai également découvert un langage de programmation, le VBA. Ce stage m'a donc permis de découvrir de nouveaux domaines et un langage que l'on étudie en cours. J'ai également découvert comment travaille une équipe dans une entreprise conséquente, mon précédent stage s'étant déroulé dans une entreprise de 15 personnes où l'esprit d'équipe n'était pas omniprésent. À Clemessy, chaque employé est prêt à aider un collègue s'il en a les compétences, ce qui n'était pas le cas dans mon autre entreprise où chaque employé était beaucoup plus personnel.

## Conclusion

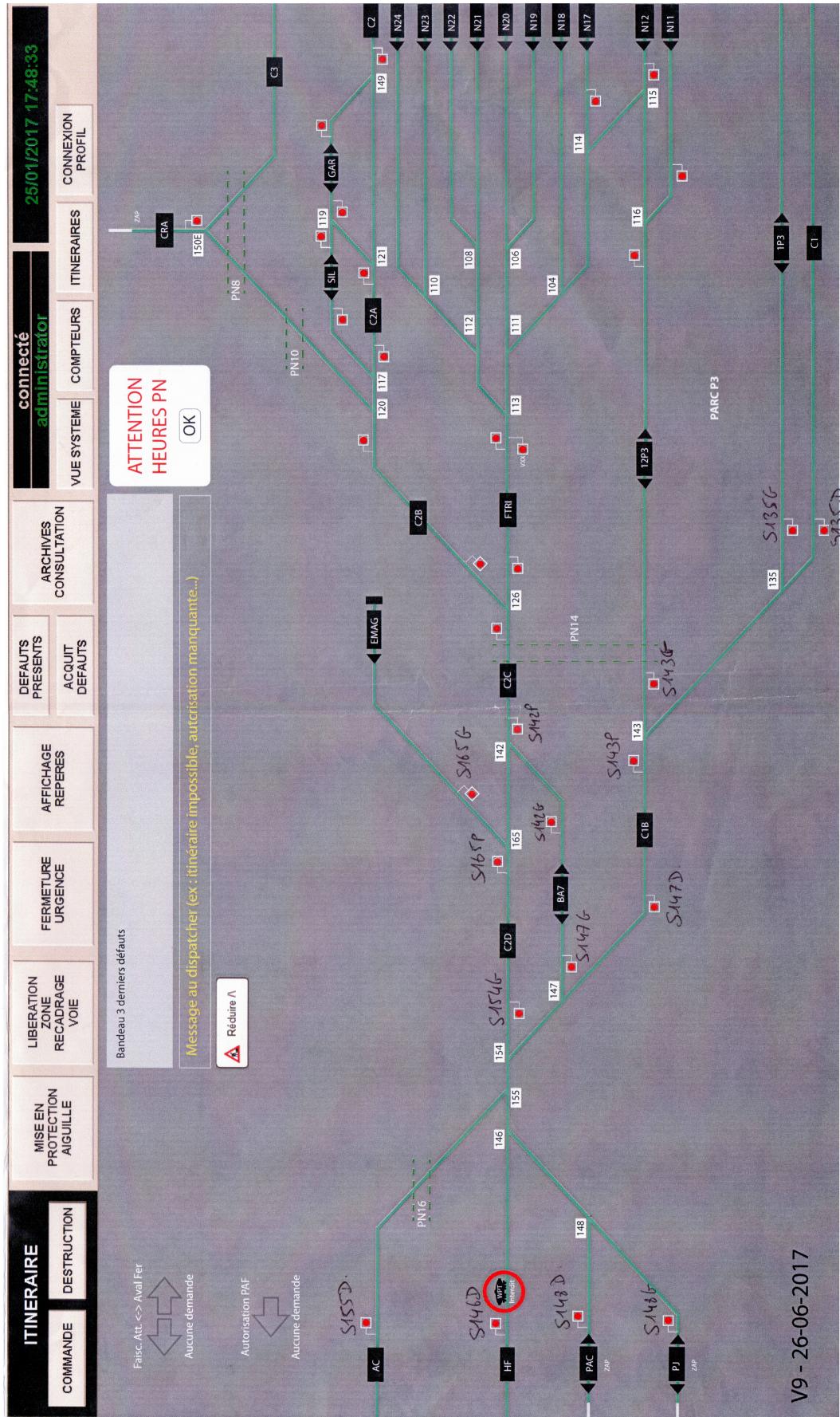
J'ai effectué mon stage de deuxième année du cycle ingénieur dans une entreprise proche de chez moi, à l'inverse de mon précédent stage à l'étranger qui s'était déroulé en Suisse. Mon objectif principal était de découvrir le monde de la supervision, ce qui a été atteint. Grâce à mes activités, j'ai pu approcher le métier d'ingénieur et ainsi le découvrir en entreprise.

Pouvoir faire partie d'une équipe constituée de plusieurs métiers m'a permis de découvrir les spécificités et la complexité d'un projet d'une grande entreprise. Grâce à cette équipe, j'ai pu acquérir de nouvelles connaissances et compétences en supervision et gestion de projet. Ces acquis me permettent de compléter les connaissances déjà acquises pour ma vie professionnelle future.

Mon expérience avec Clemessy pourra continuer en troisième année de cycle ingénieur grâce au stage ingénieur. J'ai la possibilité de faire mon stage de six mois dans l'entreprise afin d'approfondir le projet ou travailler sur un autre projet afin de pouvoir assister aux phases suivantes et plus particulièrement à celle des recettes.

## Annexes

## Annexe 1 : Modèle de la vue générale du secteur Avalfer



## Annexe 2 : fichier FC50test.AWL

```
FUNCTION "GESTION_A_158" : VOID
TITLE =FONCTION DE GESTION DE L'AIGUILLE A158
VERSION : 3.2

VAR_TEMP
    A_Pos_Droite : BOOL ;
    //Variable tampon du bit "Retour de position droite de l'aiguille"
    A_Pos_Gauche : BOOL ;
    //Variable tampon du bit "Retour de position gauche de l'aiguille"
    A_Contacteur_Aiguille : BOOL ;
    //Variable tampon du bit "Retour contacteur puissance gauche-droite-validation"
    A_Pos_Mainmoteur : BOOL ;
    //Variable tampon du bit "Retour de position du commut Mainmoteur" (1=Mainmoteur)
    A_Dmd_Auto_Droite : BOOL ;
    //Variable tampon du bit "Demande de positionnement auto a droite"
    A_Dmd_Auto_Gauche : BOOL ;
    //Variable tampon du bit "Demande de positionnement auto a gauche"
    A_Sens_Deplacement : BOOL ;
    //Variable tampon du bit "Sens du deplacement attendu" (1=Nord->Sud)
    A_Tc_Basculemt_Manu : BOOL ;
    //Variable tampon du bit "Demande de basculement manuel"
    A_Dmd_Consign_Droite : BOOL ;
    //Variable tampon du bit "Demande de positionnement a droite pour consignation"
    A_Dmd_Consign_Gauche : BOOL ;
    //Variable tampon du bit "Demande de positionnement a gauche pour consignation"
    A_Acquit_Defaut : BOOL ;
    //Variable tampon du bit "Acquittement des defauts de l'aiguille"
    A_Dmd_Raz_Cpt_Essieux : BOOL ;
    //Variable tampon du bit "Demande de remise a zero du comptage d'essieux"
    A_Dmd_Raz_Cpt_Manoeuvres : BOOL ;
    //Variable tampon du bit "Demande de remise a zero du comptage de manoeuvres"
    A_Dmd_Desappretemt : BOOL ;
    //Variable tampon du bit "Demande de desappretetement par l'aiguilleur"
    A_Inhib_Cpt_Essieux : BOOL ;
    //Variable tampon du bit "Inhibition automatique du comptage d'essieux"
    A_E_Cpt_Essieux_N1 : INT ;
    //Variable tampon de la valeur "Comptage d'essieux du 1er capteur situe au Nord"
    A_E_Cpt_Essieux_N2 : INT ;
    //Variable tampon de la valeur "Comptage d'essieux du 2nd capteur situe au Nord"
    A_E_Cpt_Essieux_S1 : INT ;
    //Variable tampon de la valeur "Comptage d'essieux du 1er capteur situe au Sud"
    A_E_Cpt_Essieux_S2 : INT ;
    //Variable tampon de la valeur "Comptage d'essieux du 2nd capteur situe au Sud"
    A_Preset_Disc_Cmd : TIME ;
    //Variable tampon de l'info "Valeur de preselection discordance commande"
    A_Preset_Disc_Pos_Droite : TIME ;
    //Variable tampon de l'info "Valeur de preselection discordance position droite"
    A_Preset_Disc_Pos_Gauche : TIME ;
    //Variable tampon de l'info "Valeur de preselection discordance position gauche"
    CPT_MVT : INT ;
END_VAR
BEGIN
```

```
NETWORK
TITLE =Etat du retour de position droite de l'aiguille
//ATTENTION : PENSER A RETIRER LE BIT DB200.DBX2.0 APRES PHASE SIMULATIONS

    U      "KAG_158_D";
    =      #A_Pos_Droite;

NETWORK
TITLE =Etat du retour de position gauche de l'aiguille
//ATTENTION : PENSER A RETIRER LE BIT DB200.DBX2.1 APRES PHASE SIMULATIONS
    U      "KAG_158_G";
    =      #A_Pos_Gauche;

NETWORK
TITLE =Etat du retour du contacteur de puissance de l'aiguille
//ATTENTION : PENSER A RETIRER LES BITS A32.4 ET A32.5 APRES PHASE SIMULATIONS
    U      "R_VAL_AIG158";
    =      #A>Contacteur_Aiguille;

NETWORK
TITLE =Etat du retour de position du mode Mainmoteur de l'aiguille

    UN     "RMM_158";
    =      #A_Pos_Mainmoteur;

NETWORK
TITLE =Etat de la demande de positionnement auto a droite de l'aiguille

    U      "PA_AIGA_C_A158_DR";
    =      #A_Dmd_Auto_Droite;

NETWORK
TITLE =Etat de la demande de positionnement auto a gauche de l'aiguille

    U      "PA_AIGA_C_A158_GA";
    =      #A_Dmd_Auto_Gauche;

NETWORK
TITLE =Sens du deplacement attendu sur l'aiguille

    U      "DATAS_CREATION_ITI".DMD_SENS_ITI;
    =      #A_Sens_Deplacement;

NETWORK
TITLE =Etat de la demande de basculement manu de l'aiguille

    U      "PA_AIGA_C_A158_AIG";
    =      #A_Tc_Basculemt_Manu;

NETWORK
TITLE =Etat de la demande de position a droite aiguille pr consignation

    U      "PA_AIGA_C_A158_CONS_DR";
    =      #A_Dmd_Consign_Droite;

NETWORK
TITLE =Etat de la demande de position a gauche aiguille pr consignation

    U      "PA_AIGA_C_A158_CONS_GA";
    =      #A_Dmd_Consign_Gauche;

NETWORK
```

```

TITLE =Etat de la demande d'acquittement des defauts

U      "Acquit_Defauts";
=      #A_Acquit_Defaut;

NETWORK

TITLE =Etat de la demande de remise a zero du comptage d'essieux

O      "PA_AIGA_C_A158_RAZ_CPT";
O      "MEMOIRE_RESET_CE".Demande_raz_aig1;
=      #A_Dmd_Raz_Cpt_Essieux;

NETWORK

TITLE =Etat de la demande de remise a zero du comptage de manoeuvres

U      "PA_AIGA_C_A158_RAZ_MNV";
=      #A_Dmd_Raz_Cpt_Manoeuvres;

NETWORK

TITLE =Etat de la demande de desappretement par l'aiguilleur

O      "PA_AIGA_C_A158_DESAPPT";
O      "COMM_RECEPTION".TC_DESAPPRETEMENT_A03;
=      #A_Dmd_Desappretemt;

NETWORK

TITLE =Etat de l'inhibition automatique du comptage d'essieux

O      "CE_158P".Demande_RAZ;
O      "CE_158G".Demande_RAZ;
O      "CE_158D".Demande_RAZ;
=      #A_Inhib_Cpt_Essieux;

NETWORK

TITLE =Affectation comptages d'essieux des capteurs Nord de l'aiguille

U(    ;
L      "CE_158G".Valeur_Comptage;
T      #A_E_Cpt_Essieux_N1;
SET   ;
SAVE  ;
CLR   ;
U     BIE;
)     ;
SPBNB _001;
L      "CE_158D".Valeur_Comptage;
T      #A_E_Cpt_Essieux_N2;
_001: NOP  0;
NETWORK

TITLE =Affectation comptages d'essieux des capteurs Sud de l'aiguille

U(    ;
L      "CE_158P".Valeur_Comptage;
T      #A_E_Cpt_Essieux_S1;
SET   ;
SAVE  ;
CLR   ;
U     BIE;

```

```

)      ;
SPBNB _002;
L      0;
T      #A_E_Cpt_Essieux_S2;
_002: NOP  0;
NETWORK
TITLE =Valeur de preselection de la tempo discordance commande

U(      ;
L      #A_Preset_Disc_Cmd;
L      T#3S;
<>D  ;
)      ;
SPBNB _003;
L      T#3S;
T      #A_Preset_Disc_Cmd;
_003: NOP  0;
NETWORK
TITLE =Valeur de preselection de la tempo discordance position droite

U(      ;
L      #A_Preset_Disc_Pos_Droite;
L      T#6S;
<>D  ;
)      ;
SPBNB _004;
L      T#6S;
T      #A_Preset_Disc_Pos_Droite;
_004: NOP  0;
NETWORK
TITLE =Valeur de preselection de la tempo discordance position gauche

U(      ;
L      #A_Preset_Disc_Pos_Gauche;
L      T#6S;
<>D  ;
)      ;
SPBNB _005;
L      T#6S;
T      #A_Preset_Disc_Pos_Gauche;
_005: NOP  0;
NETWORK
TITLE =Appel du bloc fonctionnel de l'aiguille

CALL "BLOC_AIGUILLE" , "A_158" (
    Pos_Droite          := #A_Pos_Droite,
    Pos_Gauche          := #A_Pos_Gauche,
    Contacteur_Aiguille := #A>Contacteur_Aiguille,
    Pos_Mainmoteur      := #A_Pos_Mainmoteur,
    Dmd_Auto_Droite     := #A_Dmd_Auto_Droite,
    Dmd_Auto_Gauche     := #A_Dmd_Auto_Gauche,
    Sens_Deplacement    := #A_Sens_Deplacement,

```

```

Tc_Basculemt_Manu      := #A_Tc_Basculemt_Manu,
Dmd_Consign_Droite     := #A_Dmd_Consign_Droite,
Dmd_Consign_Gauche      := #A_Dmd_Consign_Gauche,
Acquit_Defaut          := #A_Acquit_Defaut,
Dmd_RAZ_Cpt_Essieux    := #A_Dmd_Raz_Cpt_Essieux,
Dmd_RAZ_Cpt_Manoeuvre  := #A_Dmd_Raz_Cpt_Manoeuvres,
Dmd_Desappretemt        := #A_Dmd_Desappretemt,
Inhib_Cpt_Essieux       := #A_Inhib_Cpt_Essieux,
E_Cpt_Essieux_N1         := #A_E_Cpt_Essieux_N1,
E_Cpt_Essieux_N2         := #A_E_Cpt_Essieux_N2,
E_Cpt_Essieux_S1         := #A_E_Cpt_Essieux_S1,
E_Cpt_Essieux_S2         := #A_E_Cpt_Essieux_S2,
Preset_Disc_Cmd          := #A_Preset_Disc_Cmd,
Preset_Disc_Pos_Droite   := #A_Preset_Disc_Pos_Droite,
Preset_Disc_Pos_Gauche    := #A_Preset_Disc_Pos_Gauche);

NOP 0;

```

## NETWORK

TITLE =Collecte des informations issues de l'aiguille

```

CALL "COLLECTE_INFOS_A" (
  Num_DB_A           := 50,
  Commande_Droite    := "BIT_CDE_A158_DR",
  Commande_Gauche     := "BIT_CDE_A158_GA",
  Commande_Validation  := "cde_VAL_AIG158",
  Etat_Controllee_Droite := "COMM_EMISSION".TS_CONTROLEE_DR_AIG_1,
  Etat_Controllee_Gauche  := "COMM_EMISSION".TS_CONTROLEE_GA_AIG_1,
  Etat_Commandee_Droite := "COMM_EMISSION".TS_COMMANDEE_DR_AIG_1,
  Etat_Commandee_Gauche  := "COMM_EMISSION".TS_COMMANDEE_GA_AIG_1,
  Etat_Memo_Cmd_Droite  := "COMM_EMISSION".TS_MEMO_CMD_DR_AIG_1,
  Etat_Memo_Cmd_Gauche   := "COMM_EMISSION".TS_MEMO_CMD_GA_AIG_1,
  Etat_Pos_Mainmoteur   := "COMM_EMISSION".TS_POS_MAINMOTEUR_AIG_1,
  Cpt_Essieux          := "COMM_EMISSION".TM_CPT_ESSIEUX_AIG_1,
  Cpt_Manoeuvres        := #CPT_MVT,
  Defaut_Disc_Cmd        := "COMM_EMISSION".TS_DISC_CMD_AIG_1,
  Defaut_Disc_Pos        := "COMM_EMISSION".TS_DISC_POS_AIG_1);

```

## NETWORK

TITLE =Tempo mise à jour des commandes aiguilles

```

U      "cde_VAL_AIG158";
=      L      24.0;
U      L      24.0;
L      S5T#500MS;
SE    "T_SORTIES_A158";
NOP  0;
NOP  0;
NOP  0;
NOP  0;

```

```
U      L      24.0;
L      S5T#5S;
SE    "T_FIN_CMD_SORTIES_A158";
NOP   0;
NOP   0;
NOP   0;
NOP   0;

NETWORK
TITLE =Mise à jour commande droite

U      "BIT_CDE_A158_DR";
U      "cde_VAL_AIG158";
U      "T_SORTIES_A158";
UN    "T_FIN_CMD_SORTIES_A158";
=     "Cag_158_D";

NETWORK
TITLE =Mise à jour commande gauche

U      "BIT_CDE_A158_GA";
U      "cde_VAL_AIG158";
U      "T_SORTIES_A158";
UN    "T_FIN_CMD_SORTIES_A158";
=     "Cag_158_G";

END_FUNCTION
```

## Bibliographie/Sitographie

**Explication pour la tâche sur Raspberry Pi 3 :** <https://incenp.org/notes/2014/disable-new-usb-input-devices.html>

**Aide pour le VBA :** <https://www.excel-pratique.com/fr/vba.php>

**Cours de VBA :** <http://bidou.developpez.com/article/VBA/>

**Site de Clemessy :** <http://fr.clemessy.com>